NIST Special Publication 500–236

Computer Systems Technology

U.S. DEPARTMENT OF COMMERCE
Technology Administration
National Institute of Standards and Technology

NIST

The Fourth Text REtrieval Conference (TREC-4)

D. K. Harman, Editor

NIST Special Publication 500–236

The Fourth Text REtrieval Conference (TREC-4)

D. K. Harman, Editor

Computer Systems Laboratory
National Institute of Standards and Technology
Gaithersburg, MD 20899-0001

October 1996

DEPARTMENT OF COMMERCE ★ UNITED STATES OF AMERICA ★

U.S. Department of Commerce
Michael Kantor, Secretary

Technology Administration
Mary L. Good, Under Secretary for Technology

National Institute of Standards and Technology
Arati Prabhakar, Director

Reports on Computer Systems Technology

The National Institute of Standards and Technology (NIST) has a unique responsibility for computer systems technology within the Federal government. NIST's Computer Systems Laboratory (CSL) develops standards and guidelines, provides technical assistance, and conducts research for computers and related telecommunications systems to achieve more effective utilization of Federal information technology resources. CSL's responsibilities include development of technical, management, physical, and administrative standards and guidelines for the cost-effective security and privacy of sensitive unclassified information processed in Federal computers. CSL assists agencies in developing security plans and in improving computer security awareness training. This Special Publication 500 series reports CSL research and guidelines to Federal agencies as well as to organizations in industry, government, and academia.

National Institute of Standards and Technology Special Publication 500-236
Natl. Inst. Stand. Technol. Spec. Publ. 500-236, 773 pages (Oct. 1996)
CODEN: NSPUE2

U.S. GOVERNMENT PRINTING OFFICE
WASHINGTON: 1996

For sale by the Superintendent of Documents, U.S. Government Printing Office, Washington, DC 20402

Foreword

This report constitutes the proceedings of the fourth Text REtrieval Conference (TREC-4) held in Gaithersburg, Maryland, November 1-3, 1995. The conference was co-sponsored by the National Institute of Standards and Technology (NIST) and the Defense Advanced Research Projects Agency (DARPA), and was attended by 140 people involved in the 36 participating groups. The conference was the fourth in an on-going series of workshops to evaluate new technologies in text retrieval.

The workshop included plenary sessions, discussion groups and demonstrations. Because the participants in the workshop drew on their personal experiences, they sometimes cited specific vendors and commercial products. The inclusion or omission of a particular company or product does not imply either endorsement or criticism by NIST.

The sponsorship of the Intelligent Systems Office of the Defense Advanced Research Projects Agency is gratefully acknowledged, along with the tremendous work of the program committee.

Donna Harman
September 10, 1996

TREC-4 Program Committee

Donna Harman, NIST, chair
Nick Belkin, Rutgers University
Chris Buckley, Cornell University
Jamie Callan, University of Massachusetts at Amherst
Susan Dumais, Bellcore
Darryl Howard, U.S. Department of Defense
David Lewis, AT & T Bell Labs
Matt Mettler, Paracel, Inc.
John Prange, U.S. Department of Defense
Steve Robertson, City University, UK
Alan Smeaton, Dublin City University, Ireland
Karen Sparck Jones, Cambridge University, UK
Richard Tong, Sageware, Inc.
Howard Turtle, West Publishing
Ross Wilkinson, Royal Melbourne Institute of Technology

TABLE OF CONTENTS

PAPERS

APPENDICES

Abstract

This report constitutes the proceedings of the fourth Text REtrieval Conference (TREC-4) held in Gaithersburg, Maryland, November 1-3, 1995. The conference was co-sponsored by the National Institute of Standards and Technology (NIST) and the Defense Advanced Research Projects Agency (DARPA), and was attended by 140 people involved in the 36 participating groups.

The goal of the conference was to bring research groups together to discuss their work on a large test collection. There was a wide variation of retrieval techniques reported on, including methods using automatic thesaurii, sophisticated term weighting, natural language techniques, relevance feedback, and advanced pattern matching. As results had been run through a common evaluation package, groups were able to compare the effectiveness of different techniques, and discuss how differences between the systems affected performance. In addition to the main evaluation, 5 more focussed evaluations, called "tracks" were run.

The conference included paper sessions and discussion groups. This proceedings includes papers from most of the participants (several poster groups did not submit papers), tables of the system results, and brief system descriptions including timing and storage information.

Overview of the Fourth Text REtrieval Conference (TREC-4)

Donna Harman

National Institute of Standards and Technology
Gaithersburg, MD. 20899

1. INTRODUCTION

The fourth Text REtrieval Conference (TREC-4) was held at the National Institute of Standards and Technology (NIST) in November 1995. The conference, co-sponsored by DARPA and NIST, is run as a workshop for participating groups to discuss their system results on the retrieval tasks done using the TIPSTER/TREC collection. As with the first three TRECs, the goals of this workshop are:

- To encourage research in text retrieval based on large-scale test collections
- To increase communication among industry, academia and government by creating an open forum for exchange of research ideas
- To speed the transfer of technology from research labs into commercial products by demonstrating substantial improvements in retrieval methodologies on real-world problems
- To increase the availability of appropriate evaluation techniques for use by industry and academia, including development of new evaluation techniques more applicable to current systems

The number of participating systems has grown from 25 in TREC-1 to 36 in TREC-4 (see Table 1), including most of the major text retrieval software companies and most of the universities doing research in text retrieval. The diversity of the participating groups has ensured that TREC represents many different approaches to text retrieval, while the emphasis on individual experiments evaluated within a common setting has proven to be a major strength of TREC.

The research done by the participating groups in the four TREC conferences has varied, but has followed a general pattern. TREC-1 (1992) required significant system rebuilding by most groups due to the huge increase in the size of the document collection from a traditional test collection of several megabytes in size to the 2 gigabyte TIPSTER collection. The second TREC conference (TREC-2) occurred in August of 1993, less than 10 months after the first conference. The results (using new test topics) showed significant improvements over the TREC-1 results, but should be viewed as an appropriate baseline representing the 1993 state-of-the-art retrieval techniques as scaled up to a 2 gigabyte collection.

TREC-3 [Harman 1994] provided an opportunity for more complex experimentation. The experiments included the development of automatic query expansion techniques, the use of passages or subdocuments to increase the precision of retrieval results, and the use of training information to select only the best terms for queries. Some groups explored hybrid approaches (such as the use of the Rocchio methodology in systems not using a vector space model), and others tried approaches that were radically different from their original approaches. For example, experiments in manual query expansion were done by the University of California at Berkeley and experiments in combining information from three very different retrieval techniques were done by the Swiss Federal Institute of Technology (ETH).

TREC-4 represented a continuation of many of these complex experiments, and also included a set of five focussed tasks, called tracks. This paper provides a review of the TREC-4 tasks, a very brief description of the test collection being used, and an overview of the results. The papers from the individual groups should be referred to for more details on specific system approaches.

2. THE TASKS

2.1 The Main Tasks

All four TREC conferences have centered around two main tasks based on traditional information retrieval modes: a routing task and an adhoc* task. In the routing task it is assumed that the same questions are always being asked, but that new data is being searched. This task is similar to that done by news clipping services or by library profiling systems. In the adhoc task, it is assumed that new questions are being asked against a static set of data. This task is similar to how a researcher might use a

* spelled as a single word in TREC

Australian National University
CITRI, Australia
Cornell University
Dublin City University
FS Consulting
George Mason University
HNC, Inc.
InText Systems (Australia)
Logicon Operating Systems
NEC Corporation
Oracle Corporation
Rutgers University (two groups)
Swiss Federal Institute of Technology (ETH)
University of California - Berkeley
University of Central Florida
University of Kansas
University of Toronto
University of Waterloo
CLARITECH/Carnegie Mellon University
City University, London
Department of Defense
Excalibur Technologies, Inc.
GE Corporate R & D/New York University
Georgia Institute of Technology
Information Technology Institute
Lexis-Nexis
National University of Singapore
New Mexico State University
Queens College, CUNY
Siemens Corporate Research Inc.
Universite de Neuchatel
University of California - Los Angeles
University of Glasgow
University of Massachusetts at Amherst
University of Virginia
Xerox Palo Alto Research Center

Table 1: TREC-4 Participants

library, where the collection is known, but where the questions likely to be asked are unknown.

In TREC the routing task is represented by using known topics and known relevant documents for those topics, but new data for testing. The training for this task is shown in the left-hand column of Figure 1. The participants are given a set of known (or training) topics, along with a set of documents, including known relevant documents for those topics. The topics consist of natural language text describing a user's information need (see sec. 3.3 for details). The topics are used to create a set of queries (the actual input to the retrieval system) which are then used against the training documents. This is represented by Q1 in the diagram. Many sets of Q1 queries might be built to help adjust systems to this task, to create better weighting algorithms, and in general to prepare the system for testing. The results of this training are used to create Q2, the routing queries to be used against the test documents (testing task shown on the middle column of Fig. 1).

The 50 routing topics for testing are a specific subset of the training topics (selected by NIST). A new methodology was used in TREC-4 to select the routing topics and test data. Because of difficulty in getting new data, it was decided to select the new data first, and then select topics that matched the data. The ready availability of more Federal Register documents suggested the use of topics that tended to find relevant documents in the Federal Register. Twenty-five of the routing topics were picked using this criteria. This also created a subcollection of the longer, more structured Federal Register documents for later use by the research community. The second set of 25 routing topics was selected to build a subcollection in the domain of computers. The testing documents for the computer issues were documents from the Internet, plus part of the Ziff collection.

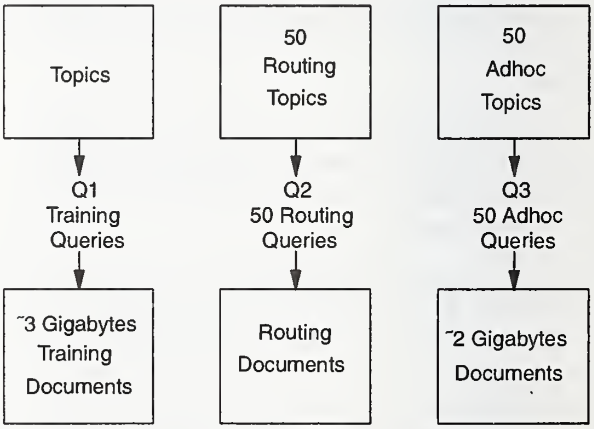


Figure 1. TREC Main Tasks.

The adhoc task is represented by new topics for known documents. This task is shown on the right-hand side of Figure 1, where the 50 new test topics are used to create Q3 as the adhoc queries for searching against the known documents. Fifty new topics (numbers 201-250) were generated for TREC-4. The known documents used in TREC-4 were on disks 2 and 3. Sections 3.2 and 3.3 give more details about the documents used and the topics that were created. The results from searches using Q2 and Q3 are the official test results sent to NIST for the routing and adhoc tasks.

In addition to clearly defining the tasks, other guidelines are provided in TREC. These guidelines deal with the methods of indexing and knowledgebase construction and with the methods of generating the queries from the supplied topics. In general, the guidelines are constructed to reflect an actual operational environment, and to allow as fair as possible separation among the diverse query construction approaches. Two generic categories of query construction were defined in TREC-4, based on the amount and kind of manual intervention used.

1. Automatic (completely automatic query construction)
2. Manual (manual query construction)

The participants were able to choose between two levels of participation: Category A, full participation, or Category B, full participation using a reduced dataset (1/4 of the full document set). Each participating group was provided the data and asked to turn in either one or two sets of results for each topic. When two sets of results were sent, they could be made using different methods of creating queries, or different methods of searching these queries. Groups could choose to do the routing task, the adhoc task, or both, and were asked to submit the top 1000 documents retrieved for each topic for evaluation.

2.2 The Tracks

One of the goals of TREC is to provide a common task evaluation that allows cross-system comparisons. This has proven to be a key strength in TREC. The second major strength is the loose definition of the two main tasks allowing a wide range of experiments. The addition of secondary tasks (tracks) in TREC-4 combines these strengths by creating a common evaluation for tasks that are either related to the main tasks, or are a more focussed implementation of those tasks.

Five formal tracks were run in TREC-4: a multilingual track, an interactive track, a database merging track, a "confusion" track, and a filtering track.

The multilingual track represents an extension of the adhoc task to a second language (Spanish). An informal Spanish test was run in TREC-3, but the data arrived late and few groups were able to take part. In TREC-4 the track was made official and 10 groups took part. There were about 200 megabytes of Spanish data (the *El Norte* newspaper from Monterey, Mexico), and 25 topics. Groups used the adhoc task guidelines, and submitted the top 1000 documents retrieved for each of the 25 Spanish topics.

The interactive track focusses the adhoc task on the process of doing searches interactively. It was felt by many groups that TREC uses evaluation for a batch retrieval environment rather than the more common interactive environments seen today. However there are few tools for evaluating interactive systems, and none that seem appropriate to TREC. The interactive track has a double goal of developing better methodologies for interactive evaluation and investigating in depth how users search the TREC topics. Eleven groups took part in this track in TREC-4. A subset of the adhoc topics was used, and many different types of experiments were run. The common thread was that all groups used the same topics, performed the same task(s), and recorded the same information about how the searches were done. Task 1 was to retrieve as many relevant documents as possible within a certain timeframe. Task 2 was to construct the best query possible.

The database merging task also represents a focussing of the adhoc task. In this case the goal was to investigate techniques for merging results from the various TREC subcollections (as opposed to treating the collections as a single entity). There were 10 subcollections defined corresponding to the various dates of the data, i.e., the three different years of the *Wall Street Journal*, the two different years of the *AP* newswire, the two sets of Ziff documents (one on each disk), and the three single subcollections (the *Federal Register*, the *San Jose Mercury News*, and the U.S. Patents). The 3 participating groups ran the adhoc topics separately on each of the 10 subcollections, merged the results, and submitted these results, along with a baseline run treating the subcollections as a single collection.

The "confusion" track represents an extension of the current tasks to deal with corrupted data such as would come from OCR or speech input. The track followed the adhoc task, but using only the category B data. This data was randomly corrupted at NIST using character deletions, substitutions, and additions to create data with a 10% and 20% error rate (i.e., 10% or 20% of the characters were affected). Note that this process is neutral in that it does not model OCR or speech input. Four groups used the baseline and 10% corruption level; only two groups tried the 20% level.

The filtering track represents a variation of the routing task, and was designed to investigate concerns about the current definition of this task. It used the same topics, training documents, and test documents as the routing task. The difference was that the results submitted for the filtering runs were unranked sets of documents satisfying three "utility function" criteria. These criteria were designed to approximate a high precision run, a high recall run, and a "balanced" run. For more details on this track see the paper "The TREC-4 Filtering Track" by David Lewis (in this proceedings).

Subset of collection	WSJ (disks 1 and 2) SJMN (disk 3)	AP	ZIFF	FR (disks 1 and 2) PAT (disk 3)	DOE
Size of collection (megabytes)					
(disk 1)	270	259	245	262	186
(disk 2)	247	241	178	211	
(disk 3)	290	242	349	245	
Number of records					
(disk 1)	98,732	84,678	75,180	25,960	226,087
(disk 2)	74,520	79,919	56,920	19,860	
(disk 3)	90,257	78,321	161,021	6,711	
Median number of terms per record					
(disk 1)	182	353	181	313	82
(disk 2)	218	346	167	315	
(disk 3)	279	358	119	2896	
Average number of terms per record					
(disk 1)	329	375	412	1017	89
(disk 2)	377	370	394	1073	
(disk 3)	337	379	263	3543	

Training and Adhoc Task

Collection Source	Size in Mbytes	Terms per Record		Total Records
		Mean	Median	
Ziff (disk 3)	249	263	119	161,021
Federal Register (1994)	283	456	390	55,554
IR Digest	7	2,383	2,225	455
News Groups	237	340	235	102,598
Virtual Worlds	28	416	225	10,152

Routing Task, TREC-4

Table 2: Document Statistics

3. THE TEST COLLECTION (ENGLISH)

3.1 Introduction

Like most traditional retrieval collections, there are three distinct parts to this collection -- the documents, the questions or topics, and the relevance judgments or "right answers."

3.2 The Documents

The documents were distributed on CD-ROMs with about 1 gigabyte of data on each, compressed to fit. For TREC-4, disks 1, 2 and 3 were all available as training material (see Table 2) and disks 2 and 3 were used for the adhoc task. New data (also shown in Table 2) was used for the routing task. The following shows the actual contents of each of the three CD-ROMs (disks 1, 2, and 3).

Disk 1

- WSJ -- *Wall Street Journal* (1987, 1988, 1989).
- AP -- *AP Newswire* (1989)
- ZIFF -- Articles from *Computer Select* disks (Ziff-Davis Publishing)
- FR -- *Federal Register* (1989)
- DOE -- Short abstracts from DOE publications

Disk 2

- WSJ -- *Wall Street Journal* (1990, 1991, 1992)
- AP -- *AP Newswire* (1988)

- ZIFF -- Articles from *Computer Select* disks
- FR -- *Federal Register* (1988)

Disk 3

- SJMN -- *San Jose Mercury News* (1991)
- AP -- *AP Newswire* (1990)
- ZIFF -- Articles from *Computer Select* disks
- PAT -- U.S. Patents (1993)

Table 2 shows some basic document collection statistics. Although the collection sizes are roughly equivalent in megabytes, there is a range of document lengths across collections, from very short documents (DOE) to very long (FR). Also, the range of document lengths within a collection varies. For example, the documents from the AP are similar in length, but the WSJ, the ZIFF and especially the FR documents have much wider range of lengths within their collections.

The documents are uniformly formatted into SGML, with a DTD included for each collection to allow easy parsing.

```
<DOC>
<DOCNO> WSJ880406-0090 </DOCNO>
<HL> AT&T Unveils Services to Upgrade Phone Networks Under Global Plan </HL>
<AUTHOR> Janet Guyon (WSJ Staff) </AUTHOR>
<DATELINE> NEW YORK  </DATELINE>
<TEXT>
  American Telephone & Telegraph Co. introduced the first of a new generation of phone services with broad
.
.
</TEXT>
</DOC>
```

3.3 The Topics

In designing the TREC task, there was a conscious decision made to provide "user need" statements rather than more traditional queries. Two major issues were involved in this decision. First, there was a desire to allow a wide range of query construction methods by keeping the topic (the need statement) distinct from the query (the actual text submitted to the system). The second issue was the ability to increase the amount of information available about each topic, in particular to include with each topic a clear statement of what criteria make a document relevant.

The adhoc topics used in TREC-3 reflected a slight change in direction from earliers TRECs. They were not only much shorter, but also were missing the complex structure of the earlier topics. In addition to being shorter and less complex, the TREC-3 (and TREC-4) topics were written by the same group of people that did the relevance judgments (see sec. 3.4). Specifically, each of the new topics (numbers 151-250) was developed from a genuine need for information brought in by the assessors. Each assessor constructed his/her own topics from some initial statements of interest, and performed all the relevance assessments on these topics (with a few exceptions).

Sample TREC-3 topic

<num> Number: 168
<title> Topic: Financing AMTRAK

<desc> Description:
A document will address the role of the Federal Government in financing the operation of the National Railroad Transportation Corporation (AMTRAK).

<narr> Narrative: A relevant document must provide information on the government's responsibility to make AMTRAK an economically viable entity. It could also discuss the privatization of AMTRAK as an alternative to continuing government subsidies. Documents comparing government subsidies given to air and bus transportation with those provided to AMTRAK would also be relevant.

Participants in TREC-3 felt that the topics were still too long compared with what users normally submit to operational retrieval systems. Therefore the TREC-4 topics were made even shorter. Only one field was used (i.e., there is no title field and no narrative field).

Sample TREC-4 Topic

<num> Number: 207

<desc> What are the prospects of the Quebec separatists achieving independence from the rest of Canada?

Table 3 gives the average number of terms for the adhoc topics for each of the TRECs. The averages are broken down by field (title, description, narrative, and concept), with all four fields for TREC-1 and TREC-2, no concept field in TREC-3, and only a description field in TREC-4. The counts are shown both including and excluding the 23 standard stopwords used by the NIST ZPRISE system.

	Stopwords	W/O Stopwords
TREC-1 (51-100)	149	99
title	6	5
description	44	27
narrative	71	41
concepts	28	26
TREC-2 (101-150)	178	125
title	7	7
description	47	30
narrative	87	54
concepts	37	34
TREC-3 (151-200)	119	70
title	6	6
description	30	18
narrative	83	46
TREC-4 (201-250)	16	10
description	16	10

Table 3: Topic Lengths

Three different topic characteristics can be observed from this table. First, there is a length difference between the topics in TREC-1 and TREC-2. The narrative and concept fields are shorter on average for the TREC-1 topics, due to the presence of many short topics. The TREC-1 topics were produced quickly, without guidelines, by several different people, whereas the TREC-2 topics were produced by a single person. This person constructed elaborate topics that are more standarized in length, and have longer narrative and concept fields.

Second, the TREC-3 topics are not only missing the concept fields (by design), but also contain significantly shorter description fields. The TREC-3 topics were written by the 10 TREC-3 assessors who made the relevance judgments for those topics. The types of questions being asked by these assessors were less complex than the more studied questions in TREC-2, and this resulted in shorter description fields. The narrative fields are about the same length, however, probably because the TREC-2 topics were used as an example of how to write narratives. The shorter description fields, and lack of concept fields, led to topics that are about two-thirds the length of the TREC-2 topics.

The third noticeable topic characteristic is that the TREC-4 topics are much shorter than the TREC-3 topics. Not only are the narrative fields removed, but the title field is also gone. In addition, the description fields turned out to be significantly shorter going from TREC-3 to TREC-4. This was not expected, but resulted from a change in the way the topics were built. In TREC-3 the assessors brought in "seeds" of topics, i.e., ideas of issues on which to build a topic. These seeds were then expanded by each assessor, based on looking at the items that were retrieved. The resulting topics were therefore "tuned" to the collections, and were still "artificial" topics.

To avoid this tuning in TREC-4, the assessors were asked to bring in completed topics, i.e., one-sentence descriptions that were used for the actual searching. The final set of 50 topics in TREC-4 were selected by NIST from approximately 150 of these initial searches. The selection was based on how many "relevant" documents were found during sample searching. The candidate topics that retrieved too many "relevant" documents were rejected; topics were also rejected that seemed ambiguous. This different method of constructing topics resulted in the much shorter descriptions that tended to resemble the "seeds" of the TREC-3 topics rather than the TREC-3 description section. The very short topics in TREC-4 had a major effect on the results.

3.4 The Relevance Judgments

The relevance judgments are of critical importance to a test collection. For each topic it is necessary to compile a list of relevant documents; hopefully as comprehensive a list as possible. All four TRECs have used the pooling method [Sparck Jones & van Rijsbergen 1975] to assemble the relevance assessments. In this method a pool of possible relevant documents is created by taking a sample of documents selected by the various participating systems. This sample is then shown to the human assessors. The particular sampling method used in TREC is to take the top 100 documents retrieved in each submitted run for a given topic and merge them into the pool for assessment. This is a valid sampling technique since all the systems used ranked retrieval methods, with those documents most likely to be relevant returned first.

A measure of the effect of pooling can be seen by examining the overlap of retrieved documents. Table 4 shows the statistics from the merging operations in the four TREC conferences. For example, in TREC-1 and TREC-2 the top 100 documents from each run (33 runs in TREC-1 and 40 runs in TREC-2) could have produced a total of 3300 and 4000 documents to be judged (for the adhoc task). The average number of documents actually judged per topic (those that were unique) was 1279 (39%) for TREC-1 and 1106 (28%) for TREC-2. Note that even though the number of runs increased in TREC-2 by more than 20%, the number of unique documents found actually dropped. The percentage of relevant documents found, however, has not changed much for the adhoc task. (The TREC-2 routing task had many fewer relevant documents because the topics were designed to be much narrower in scope.) The more accurate results going from TREC-1 to TREC-2 mean that fewer nonrelevant documents were being found by the systems. This trend continued in TREC-3, with a drop in the number of unique documents being found (particularly for the routing task) that reflects increased accuracy in rejecting nonrelevant documents. (Since only one run per system was judged, a higher percentage of documents were unique. Note a correction from the TREC-3 proceedings.)

	Adhoc		
	Possible	Actual	Relevant
TREC-1	3300	1279 (39%)	277 (22%)
TREC-2	4000	1106 (28%)	210 (19%)
TREC-3	2700	1005 (37%)	146 (15%)
TREC-4	7300	1711 (24%)	130 (7.5%)
adhoc	4000	1345 (34%)	115 (8.5%)
confusion	900	205	0
dbmerge	800	77	2
interactive	1600	84	13

	Routing		
	Possible	Actual	Relevant
TREC-1	2200	1067 (49%)	371 (35%)
TREC-2	4000	1466 (37%)	210 (14%)
TREC-3	2300	703 (31%)	146 (21%)
TREC-4	3800	957 (25%)	132 (14%)
routing	2600	930 (35%)	131 (14%)
filtering	1200	27	1 (14%)

Table 4: Overlap of Submitted Results

In TREC-4, however, the trend was reversed. Table 4 presents the statistics from the main tasks (adhoc and routing) and the associated tracks. Note that the numbers given for the tracks are in addition the main tasks, e.g., there was an average of 205 additional unique documents found from runs in the confusion track, but no new relevant documents were found. The numbers of unique documents are affected by the order of merging; that is, the average number of unique documents found by the interactive track does not count documents already found by the confusion and dbmerge tracks. The average number of relevant documents found per task or track is the actual average of unique relevant documents for that track or task.

In the case of the adhoc runs (including most of the track runs), there is a slight increase in the percentage of unique documents found. Looking at the adhoc task alone, a relatively high percentage of unique (mostly nonrelevant) documents were found. This is likely caused by the wide variety of expansion terms used by the systems to compensate for the lack of a narrative section in the topic. The additional unique documents found by the tracks appears to be characteristic of the type of methodology being tested within that track. The confusion track, where the data is corrupted for most of the runs, turned in many unique (and all nonrelevant) documents. The data merging track turned in far fewer unique documents, and some of these were additional relevant documents. The interactive track (the last to be merged) still found additional unique documents, with a relatively high percentage of those documents being relevant.

Slightly more unique documents were found for the routing task in TREC-4 than in TREC-3, probably resulting from the increased difficulty of the TREC-4 routing task. This increased difficulty stems from 1) the concentration of long Federal Register documents, which have consistently been harder to retrieve, and 2) a mismatch of the testing data to the training data (for the computer topics). Both these factors led to less accurate filtering of nonrelevant documents.

The total number of relevant documents found has dropped with each TREC, and that drop has been caused by a deliberate tightening of the topics each year to better guarantee completeness of the relevance judgments (see below for more details on this).

Evaluation of retrieval results using the assessments from this sampling method is based on the assumption that the vast majority of relevant documents have been found and that documents that have not been judged can be assumed to be not relevant. A test of this assumption was made using TREC-2 results, and again during the TREC-3 evaluation. In both cases, a second set of 100 documents was examined from each system, using only a sample of topics and systems in TREC-2, and using all topics and systems in TREC-3.

For the TREC-2 completeness tests, a median of 21 new relevant documents per topic was found (11% increase in total relevant documents). This averages to three new relevant documents found in the second 100 documents for each system, and this is a high estimate for all systems since the 7 runs sampled for additional judgments were from the better systems. Similar results were found for the more complete TREC-3 testing, with a median of 30 new relevant documents per topic for the adhoc task, and 13 new ones for the routing task. This averages to about one new relevant document per run, since 27 runs from all systems were used in the adhoc test (23 runs in the routing test). These levels of completeness are quite acceptable for this type of evaluation.

The number of new relevant documents found was shown to be correlated with the original number of relevant documents. Topics with many more relevant documents initially tend to have more new ones found, and this has led to a greater emphasis on using topics with fewer relevant documents.

In addition to the completeness issue, relevance judgments need to be checked for consistency. In each of the TREC evaluations, each topic was judged by a single assessor to ensure the best consistency of judgment. Some testing of this consistency was done after TREC-2, when a sample of the topics and documents was rejudged by a second assessor. The results showed an average agreement between the two judges of about 80%. In TREC-4 all the adhoc topics had samples rejudged by two additional assessors, with the results being about 72% agreement among all three judges, and 88% agreement between the initial judge and either one of the two additional judges. This is a remarkably high level of agreement in relevance assessment, and probably is due to the general lack of ambiguity in the topics. More consistency checking will be done before TREC-5, particularly investigating known inconsistencies and topics with major disagreements.

4. Evaluation

An important element of TREC is to provide a common evaluation forum. Standard recall/precision and recall/fallout figures have been calculated for each TREC system and are shown in Appendix A, along with some single evaluation measures for each system. A detailed explanation of the measures is also included in the appendix.

Additional data about each system was collected that describes system features and system timing, and allows some primitive comparison of the amount of effort needed to produce the results. The individual system descriptions are given in Appendix B.

5. Results

5.1 Introduction

One of the important goals of the TREC conferences is that the participating groups freely devise their own experiments within the TREC task. For some groups this means doing the routing and/or adhoc task with the goal of achieving high retrieval effectiveness performance. For other groups, however, the goals are more diverse and may mean experiments in efficiency, unusual ways of using the data, or experiments in how "users" would view the TREC paradigm.

The overview of the results discusses the effectiveness of the systems and analyzes some of the similarities and differences in the approaches that were taken. In all cases, readers are referred to the system papers in this proceedings for more details.

5.2 TREC-4 Adhoc Results

The TREC-4 adhoc evaluation used new topics (topics 201-250) against two disks of training documents (disks 2 and 3). Only 49 topics were used in the actual evaluation as topic 201 retrieved no relevant documents. A dominant feature of the adhoc task was the much shorter topics (see more on this in the discussion of the topics, section 3.3). Many groups tried their automatic query expansion methods on the shorter topics (with good success); other groups also did manual query construction experiments to contrast these methods for the very short topics.

There were 39 sets of results for adhoc evaluation in TREC-4, with 33 of them based on runs for the full data set. Of these, 14 used automatic construction of queries, and 19 used manual construction. All of the category B groups used automatic construction of the queries.

Figure 2 shows the recall/precision curves for the 6 TREC-4 groups with the highest non-interpolated average precision using automatic construction of queries. The runs are ranked by the average precision and only one run is shown per group (both official Cornell runs would have qualified for this set).

A short summary of the techniques used in these runs shows the breadth of the approaches and the changes in approach from TREC-3. For more details on the various runs and procedures, please see the cited papers in this proceedings.

CrnlEA -- Cornell University ("New Retrieval Approaches Using SMART: TREC-4" by Chris Buckley, Amit Singhal, Mandar Mitra, (Gerald Salton)) used the SMART system, but with a non-cosine length normalization method. The top 20 documents were used to locate 50 terms and 10 phrases for expansion, as contrasted with using the top 30 documents to massively expand (500 terms + 10 phrases) the topics as in TREC-3. This change in expansion techniques was mostly due to the major change in the basic algorithm. However, additional care was taken not to overexpand the very short topics.

pircs1 -- Queens College, CUNY ("TREC-4 Ad-Hoc, Routing Retrieval and Filtering Experiments using PIRCS" by K.L. Kwok and L. Grunfeld) used a spreading activation model on subdocuments (550-word chunks). It was expected that this type of model would be particularly affected by the shorter topics, and experiments were run trying several methods of topic expansion. For this automatic run, expansion was done by selecting 50 terms from the top 40 subdocuments in addition to the terms in the original topic. Several other experiments were made using manual modifications/expansions of the topics and these are reported with the manual adhoc results.

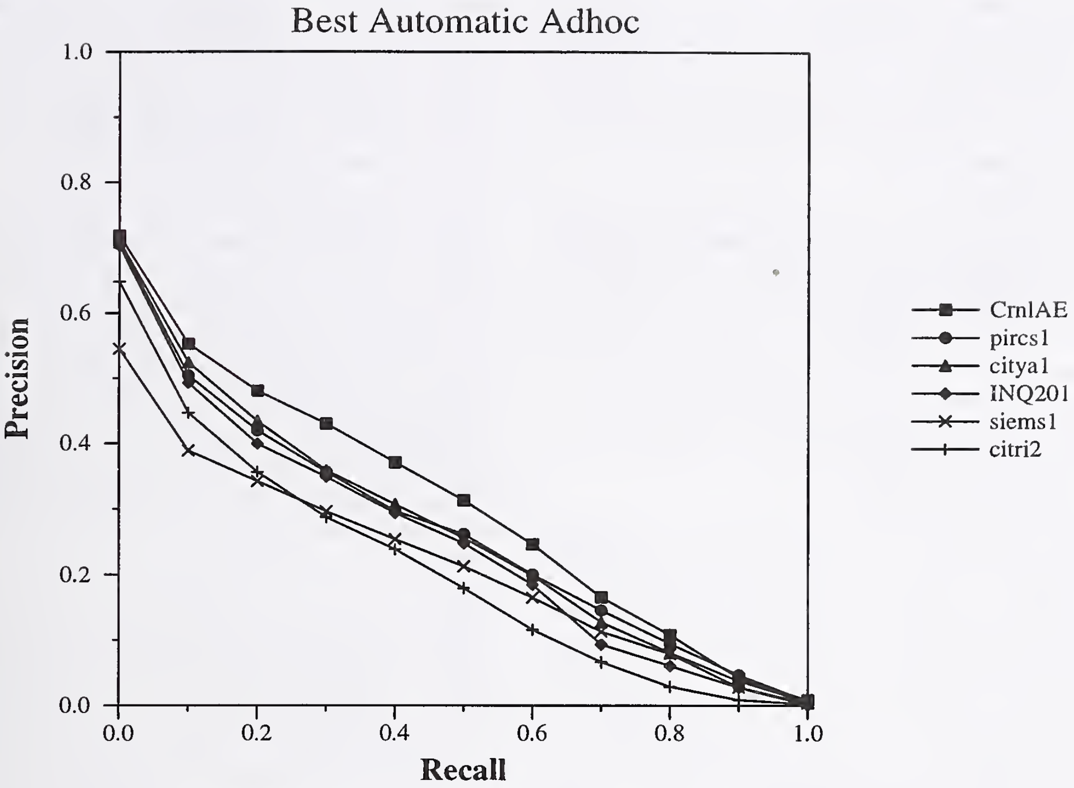


Figure 2. Best TREC-4 Automatic Adhoc Results.

citya1 -- City University, London ("Okapi at TREC-4" by S.E. Robertson, S. Walker, M.M. Beaulieu, M. Gatford and A. Payne") used a probabilistic term weighting scheme similar to that used in TREC-3. An average of 20 terms were automatically selected from the top 50 documents retrieved (only initial and final passages of these documents were used for term selection). The use of passages seemed to have little effect. This run was a base run for their experiments in manual query editing.

INQ201 -- University of Massachusetts at Amherst ("Recent Experiments with INQUERY" by James Allan, Lisa Bellesteros, James P. Callan, W. Bruce Croft and Zhihong Lu) used a version of probabilistic weighting that allows easy combining of evidence (an inference net). Their basic term weighting formula underwent a major change between TREC-3 and TREC-4 that combined the TREC-3 INQUERY weighting with the OKAPI (City University) weighting. They also used passage retrieval as in TREC-3, but found it detrimental in TREC-4. The topics were expanded by 30 phrases that were automatically selected from a phrase "thesaurus" (InFinder) that had previously been built automatically from the entire corpus of documents. Expansion did not work as well as in TREC-3.

siems1 -- Siemens Corporate Research ("Siemens TREC-4 Report: Further Experiments with Database Merging" by Ellen M. Voorhees) used the SMART retrieval strategies from TREC-3 in this run (their base run for the database merging track). The standard vector normalization was used, and query expansion was done using the Rocchio method to select up to 100 terms and 10 phrases from the top 15 documents retrieved.

citri2 -- RMIT, Australia ("Similarity Measures for Short Queries" by Ross Wilkinson, Justin Zobel, and Ron Sacks-Davis) was the result of a series of investigations into similarity measures. The best of these measures combined the standard cosine measure with the OKAPI measure. No topic expansion was done for this run.

It is interesting to note that many of the systems did critical work on their term weighting/similarity measures between TREC-3 and TREC-4. Three of the top 6 runs were results of major revisions in the basic ranking algorithms, revisions that were the outcome of extensive analysis work on previous TREC results. At Cornell they investigated the problems with using the cosine normalization on the long documents in TREC. This investigation resulted in a completely new term weighting/similarity strategy that performs well for all lengths of documents. The University of Massachusetts examined the issue of dealing with terms having a high frequency in documents (which is also related to document length).

The result of their investigation was a term weighting algorithm that combined the OKAPI algorithm (City University) for high frequency terms with the old INQUERY algorithm for lower frequency terms. The work at RMIT (the *citri2* run) was part on their ongoing effort to test various term weighting schemes.

These experiments in more sophisticated term weighting and matching algorithms are yet another step in the adaptation of retrieval systems to a full-text environment. The issues of long documents, with their higher frequency terms, mean that the algorithms originally built for abstract-length documents need rethinking. This did not happen in earlier TRECs because the problem seemed less important than, for example, discovering automatic query expansion methods in TREC-3.

The dominant new feature in TREC-4 was the very short topics. These topics were much shorter than any previous TREC topics (an average reduction from 119 terms in TREC-3 to 16 terms in TREC-4). In general the participating groups took two approaches: 1) they used roughly the same techniques that they would have on the longer topics, and 2) most of them tried some investigative manual experiments. Of the 6 runs shown in Figure 2, two runs (*INQ201* and *cityal*) used a similar number and source of expansion terms as for the longer queries. The SMART group (*CrnlAE*) used many fewer terms because of their new algorithms. The *pircs1* run was a result of more expansion, but this was due to corrections of problems in TREC-3 as opposed to changes needed for the shorter topics. The run from Siemens *siems1* was made as a baseline for database merging, and therefore had less expansion. There was no expansion in the *citri21* run.

Figure 3 shows the comparison of results between TREC-3 and TREC-4 for 4 of the groups that did well in each evaluation. As expected, all groups had worse performance. The performance for City University, where similar algorithms were used in TREC-3 and TREC-4, dropped by 36%. A similar drop (34%) was true for the INQUERY results, even though the new algorithm resulted an almost 5% improvement in results (for the TREC-4 topics). Whereas the Cornell results represented a major improvement in performance over the TREC-3 algorithms, their overall performance dropped by 14%.

This points to several issues that need further investigation in TREC-5. First, experiments must still continue on the shorter topics, since this represents the typical initial input query. The results from the shorter topics may be so poor that the top documents provide misleading expansion terms. This was a major concern in TREC-3 and analysis of this issue is clearly needed. The fact that passage retrieval, which provided substantial improvement of results in TREC-3, did not help with the shorter TREC-4 topics indicates that other types of "noise" control may be needed for short topics. It may be that the statistical "clues" presented by these shorter topics are simply not enough to provide good retrieval performance and that better human-aided systems need to be tested.

However, the manual systems also suffered major drops in performance (see Figure 4). This leads to a second issue, i.e., a need for further investigation into the causes of the generally poorer performance in the TREC-4 adhoc task. It may be that the narrative section of the topic is necessary to make the intent of the user clear to both the manual query builder and the automatic systems. The fact that machine performance mirrored human performance in TREC-4 makes the decrease in automatic system performance more acceptable, but still requires further analysis into why both types of query construction were so affected by the very short topics.

Figure 5 shows the recall/precision curves for the 6 TREC-4 groups with the highest non-interpolated average precision using manual construction of queries. A short summary of the techniques used in these runs follows. Again, for more details on the various runs and procedures, see the cited papers in this proceedings.

CnQst2 -- Excalibur Corporation ("The Excalibur TREC-4 System, Preparations and Results" by Paul E. Nelson) used manually built queries. This system uses a two-level searching scheme in which the documents are first ranked via coarse-grain methods, and then the resulting subset is further refined. There are thesaurus tools available for expansion, and this run was the result of many experiments into such issues as term groupings and assignment of term strengths.

pircs2 -- Queens College, CUNY ("TREC-4 Ad-Hoc, Routing Retrieval and Filtering Experiments using PIRCS" by K.L. Kwok and L. Grunfeld) is a manual modification of the automatic queries in *pircs1*. The modification was to replicate words (this increases the weight) and to add a few associated words (an average of 1.73 words per query or at most 3 content words). The simple replication of words led to a 12% increase in performance; adding the associated words (the *pircs2* run) upped this increase to 30% improvement over the initial automatic query.

uwgcl1 -- University of Waterloo ("Shortest Substring Ranking (MultiText Experiments for TREC-4)" by Charles L.A. Clarke, Gordon V. Cormack, and Forbes J. Burkowski) used queries that were manually built in a special query language called GCL. This query language uses Boolean operators and proximity constraints to

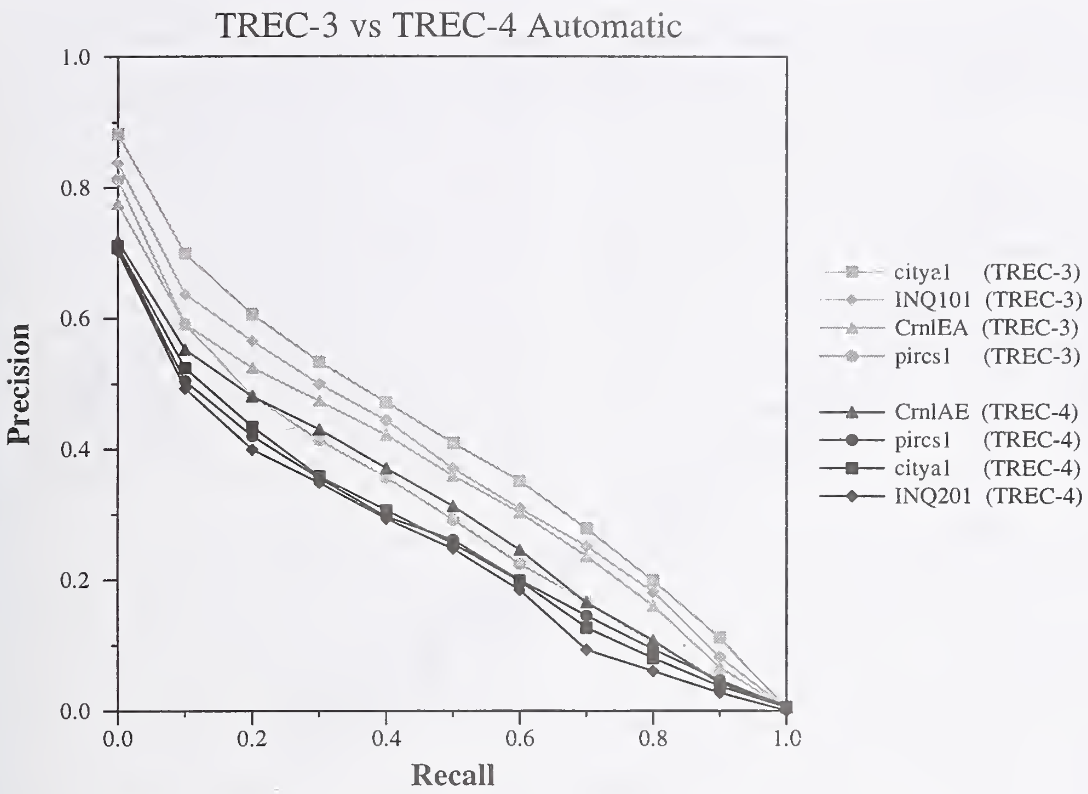


Figure 3. Comparison of Automatic Adhoc Results for TREC-3 and TREC-4.

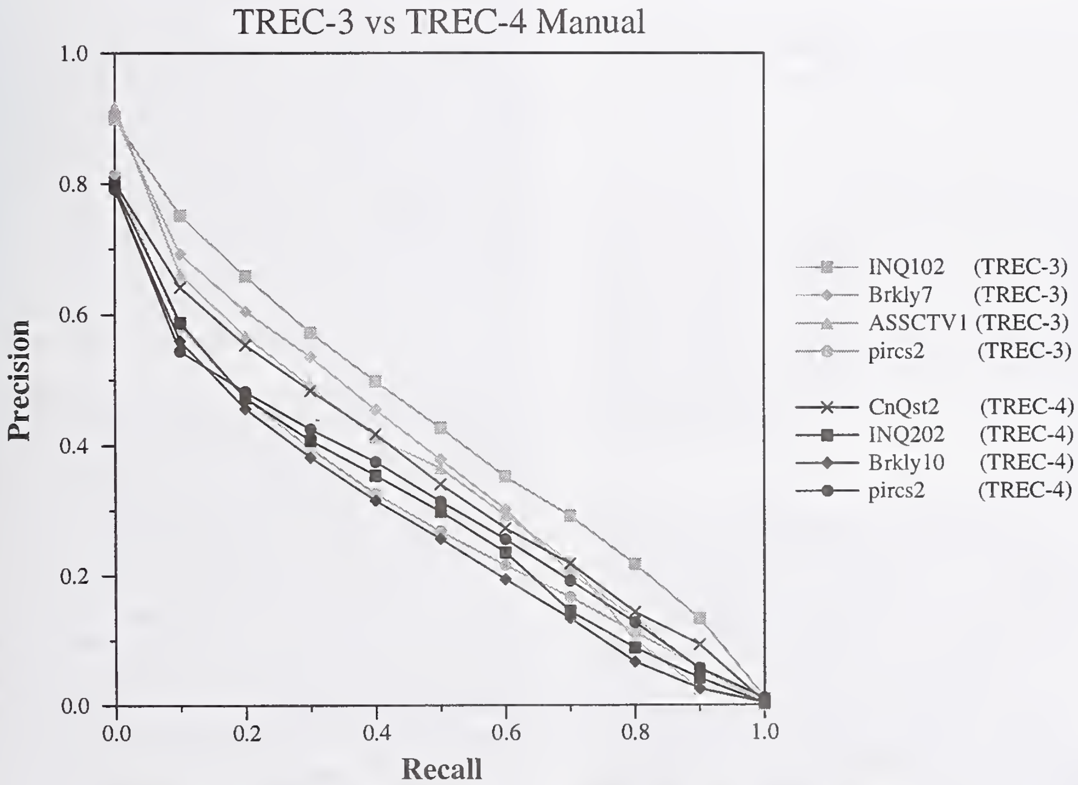


Figure 4. Comparison of Manual Adhoc Results for TREC-3 and TREC-4.

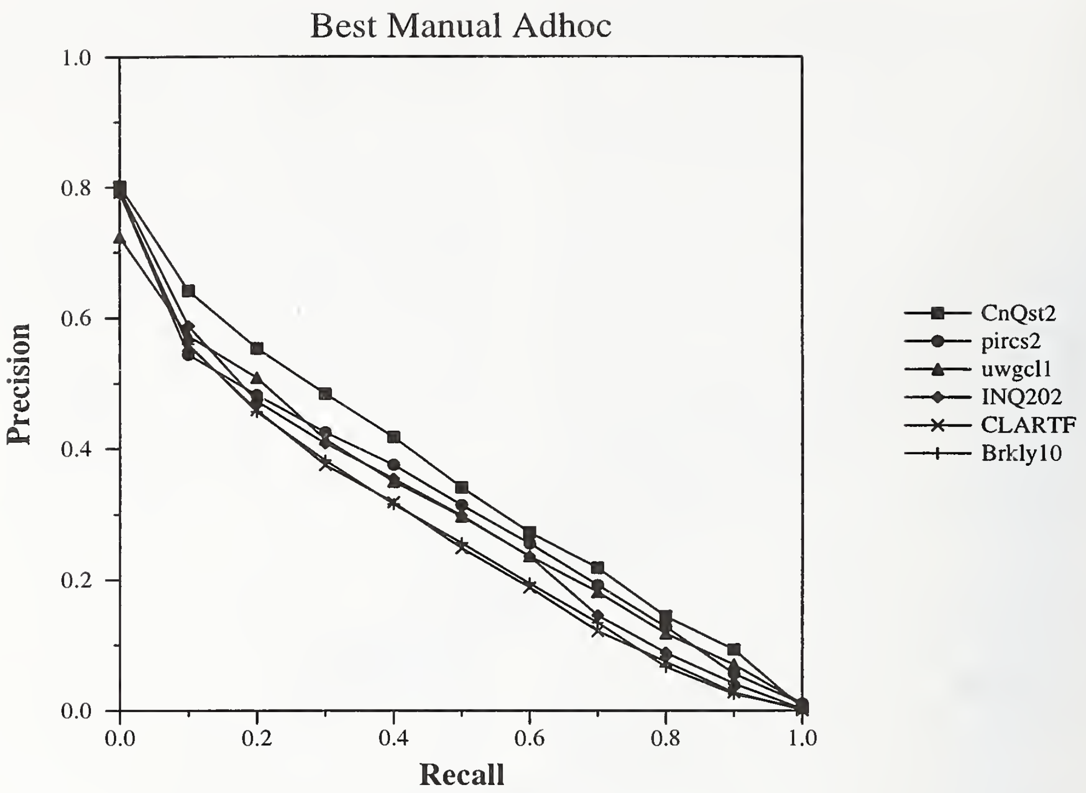


Figure 5. Best TREC-4 Manual Adhoc Results.

create intervals of text that satisfy specific conditions. The ranking algorithms rely on combining the results of increasingly less restrictive queries until the 1000 document list is created.

INQ202 -- University of Massachusetts at Amherst ("Recent Experiments with INQUERY" by James Allan, Lisa Bellesteros, James P. Callan, W. Bruce Croft and Zhihong Lu) This run is a manual modification of the *INQ201* run, with strict rules for the modifications that only allow removal of words and phrases, modification of weights, and addition of proximity restrictions. This type of manual modification increased overall average precision by 21%. The same types of modification gained only 15.5% in TREC-3.

CLARTF -- CLARITECH Corporation ("CLARIT TREC-4 Experiments" by David A. Evans, Natasa Milic-Frayling, and Robert G. Lefferts) used the CLARIT system in a machine-aided manual query construction process. The initial query terms were manually modified and weighted, and then terms were manually selected for addition to the query based on an automatic thesaurus extraction process. This particular run used a manually-built "required terms filter" to locate the best document windows for use in the thesaurus extraction process.

Brkly10 -- University of California, Berkeley ("Logistic Regression at TREC4: Probabilistic Retrieval from Full Text Document Collections" by Fredric C. Gey, Aitao Chen, Jianzhang He and Jason Meggs) used manually-reformulated queries including expansion using the News database of the MELVYL electronic catalog to either add specific instances or synonyms and related terms. The basic retrieval system is a logistic regression model that combines information from 6 measures of document relevancy based on term matches and term distribution. The coefficients were learned from the training data.

These 6 runs (and most of the other manual runs) can be divided into three different styles of manual query construction. The first group uses an automatic query construction method as a starting point, and then manually modifies the results. The *INQ202* run is a good example of this, where words and phrases were removed, term weights were modified, and proximity restrictions were added to the initial automatic query. The *pircs2* results were based on reweighting of the automatically generated terms and then adding a few new terms. The *cityml* (not shown) results were based on pre-editing the automatically generated query, and then post-editing the automatic expansion of that query.

The results of these manual modifications were highly varied. The manual edits performed by City University were only marginally effective. Manual modification of term weights seemed to have more impact, as is illustrated by the 12% improvement in the *pircs2* run, and also by some unknown percentage of the INQUERY manual results. However the addition of a few expansion terms in the *pircs2* run, or the use of proximity restrictions (*INQ202*) look to be the most promising manual modifications. Note that several of the runs in this top 6 make heavy use of some type of proximity restrictions. The ConQuest group found major improvements from term grouping, and the Multitext system from the University of Waterloo relies on proximity restrictions for their results. Since proximity restrictions are related to the use of phrases (either statistical or syntactic) or the use of additional local information, this area is clearly a focus for further research.

The second type of manual query construction, exemplified by *uwgcl1* and *Brkly10*, used queries completely manually generated using some type of auxiliary information resource such as online dictionaries (*uwgcl1*) or news databases (*Brkly10*). The query generated for *uwgcl1* uses Boolean-type restrictors, whereas the query generated for *Brkly10* uses natural language.

The third type of manual query construction involves a more complex type of human-machine interaction. Both the *CnQst2* run and the *CLARTF* run are results of experiments examining a multi-stage process of query construction. The ConQuest group starts with a manual query, and then expands this query semi-automatically by manually choosing the correct senses of terms to expand. Then they manually modify the term weights and term grouping. The CLARITECH group manually modifies queries that are automatically generated, and then provides various levels of user control of an automatic expansion process (see the CLARITECH paper for several experiments involving this user control).

Note that these three styles of manual query construction require various levels of user effort and training. Simple edits of automatic queries, user term weighting, and (less likely) proximity restrictions can be done by a relatively untrained user. The performance of these users is not apt to be as good as the *INQ202* or *pircs2* results, however, since both of these runs were the results of the primary system developers functioning as users.

The complete manual generation of queries (such as the *uwgcl1* or *Brkly10* efforts) require the types of skills currently seen in search intermediaries. Using specific query languages takes lots of training, and learning to find reasonable terms to expand topics is an art acquired only after lots of practice. This should be contrasted with the third type of query construction. The complex interaction with the user exemplified by the *CnQst2* and *CLARTF* runs requires a different type (and possibly level) of skills and training. These systems are a completely new model of search engine, and it will be necessary to develop different skills and new "mental models" in order that users can become proficient in searching.

The amount of effort and training required to achieve these improvements over automatic results should not preclude using these techniques. Indeed the major improvements shown by these methods illustrate the importance of continuing investigation into the best places for human intervention. Many studies have shown that users feel a need for more control of their searching and this control is absent from current automatic systems.

5.5 TREC-4 Routing Results

The routing evaluation used a specifically selected subset of the training topics, with that selection guided by the availability of new testing data. The ease of obtaining more Federal Register documents suggested the use of topics that tended to find relevant documents in the Federal Register and 25 of the routing topics were picked using this criterion. The second set of 25 routing topics were selected to build a subcollection in the domain of computers. The testing documents for the computer issues were documents from the Internet, plus part of the Ziff collection (see table 3).

There were a total of 28 sets of results for routing evaluation, with 26 of them based on runs for the full data set. Of the 26 systems using the full data set, 23 used automatic construction of queries, and 3 used manual construction. There were 2 sets of category B routing results, both using automatic construction of queries.

Figure 6 shows the recall/precision curves for the 6 TREC-4 groups with the highest non-interpolated average precision for the routing queries. The runs are ranked by the average precision. A short summary of the techniques used in these runs follows. For more details on the various runs and procedures, please see the cited papers in this proceedings.

INQ203 -- University of Massachusetts at Amherst ("Recent Experiments with INQUERY" by James Allan, Lisa Bellesteros, James P. Callan, W. Bruce Croft and Zhihong Lu) used the inference net engine (same as for the adhoc task), but made major refinements of the algorithms used in TREC-3. The queries were constructed using a Rocchio weighting approach for terms in relevant and non-relevant training documents, and then these queries were expanded by 250 new concepts (adjacent term pairs) found in the 200-word best-matching windows in the relevant documents. Further experiments were made in weighting terms, including use of the

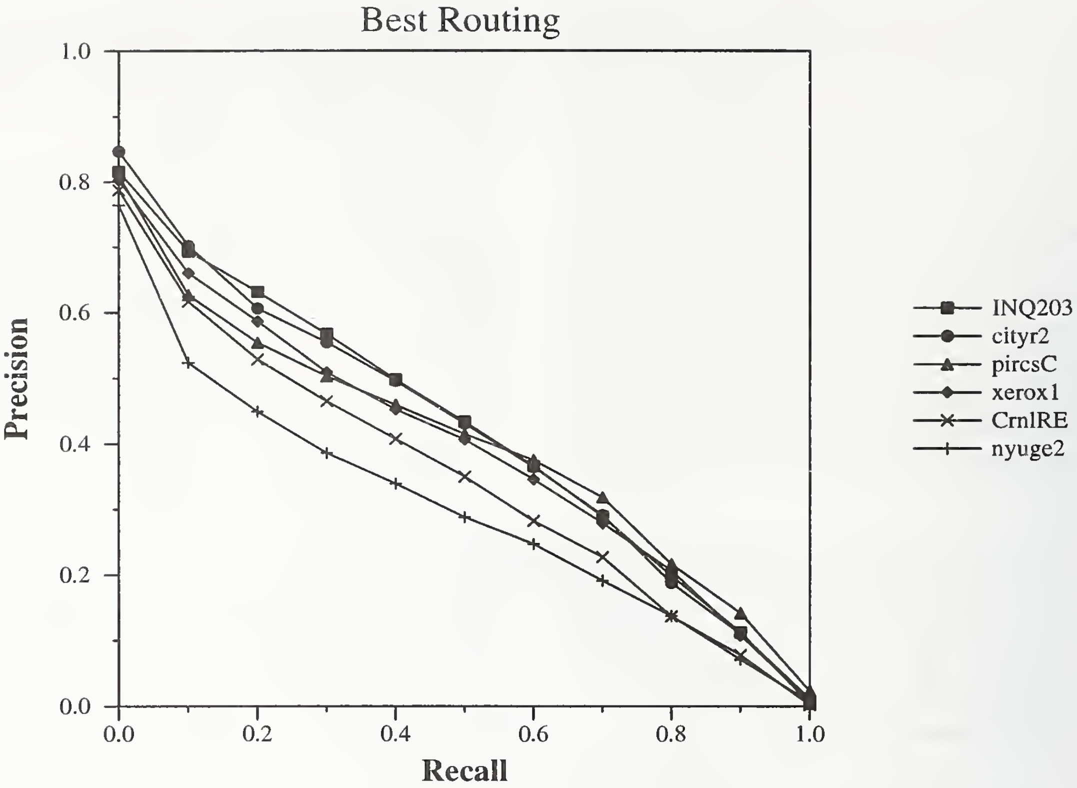


Figure 6. Best TREC-4 Routing Results.

Dynamic Feedback Optimization from Cornell (and City University).

cityr2 -- City University, London ("Okapi at TREC-4" by S.E. Robertson, S. Walker, M.M. Beaulieu, M. Gatford and A. Payne") used the same probabilistic techniques as for the adhoc task, but constructed the query using a very selective set of terms (36 on average) from the relevant documents, similar to their TREC-3 approach. The method used for term selection involved optimizing the query based on trying different combinations of terms from the relevant documents. Since this is a very compute-intensive method, the work for TREC-4 looked for more efficient methods.

pircsC -- Queens College, CUNY ("TREC-4 Ad-Hoc, Routing Retrieval and Filtering Experiments using PIRCS" by K.L. Kwok and L. Grunfeld) used the same spreading activation model used in the adhoc task, but combined the results of four different query experts. Two of these query experts used different levels of topic expansion (80 terms and 350 terms), and the other two were trained on specific subsets of the data (FR and Ziff vs WSJ, AP and SJMN).

xerox1 -- Xerox Research Center ("Xerox Site Report: Four TREC-4 Tracks" by Marti Hearst, Jan Pedersen, Peter Pirolli, Hinrich Schutze, Gregory Grefenstette and David Hull) used a complex routing algorithm that involved using LSI techniques to discover the best features, and then used three different classification techniques (combined) to rank the documents selected by these features.

CrnlRE -- Cornell University ("New Retrieval Approaches Using SMART: TREC-4" by Chris Buckley, Amit Singhal, Mandar Mitra, (Gerald Salton)) worked with the same new SMART algorithms used in the adhoc task. Because of inexperience with these new algorithms, minimal query expansion was used (only 50 single terms, as opposed to the TREC-4 300 terms). Dynamic query optimization was tried, but did not help.

nyuge2 -- GE Corporate Research and New York University ("Natural Language Information Retrieval: TREC-4 Report" by Tomek Strzalkowski and Jose Perez Carballo) used NLP techniques to discover syntactic phrases in the documents. Both single terms and phrases were indexed and specially weighted. The *nyuge2* run used topic expansion of up to 200 terms and phrases based on the relevant documents.

The issue of what features of documents should be used for retrieval was the paramount issue for all these groups (plus most of the other groups doing the routing task). It

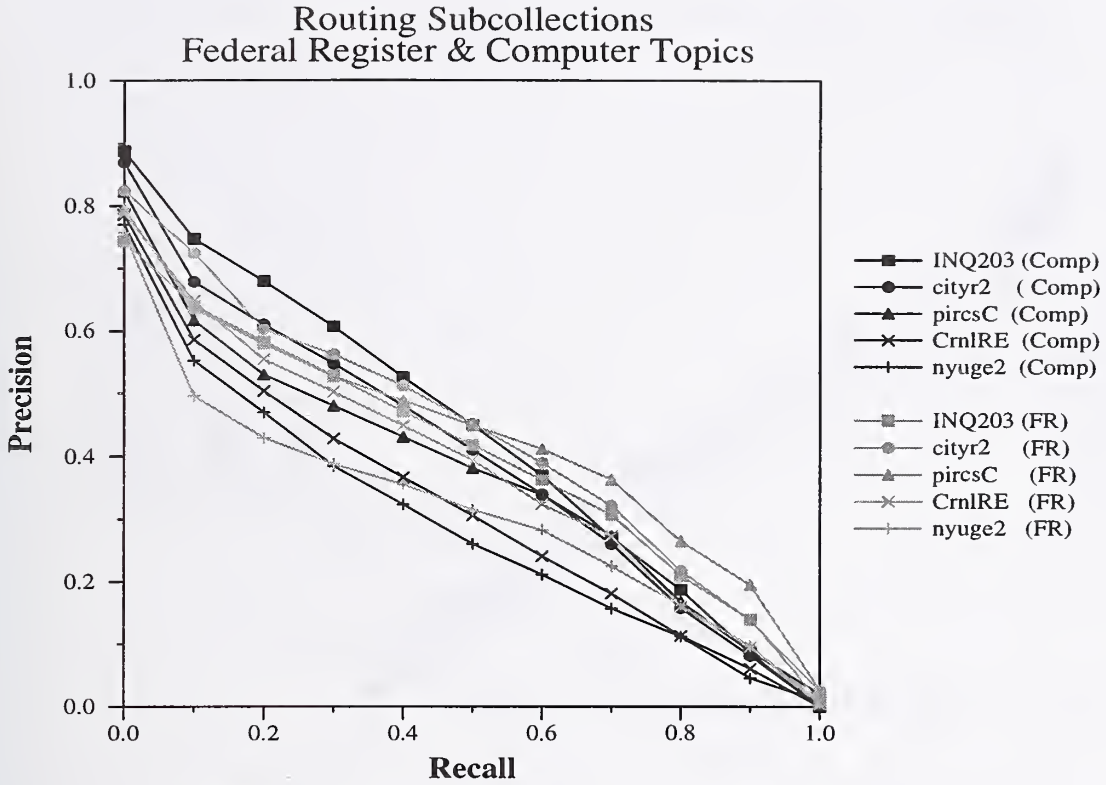


Figure 7. Comparison of Results for Federal Register Topics.

is interesting that the six groups shown in Figure 6 have used very different methods. The Cornell group used traditional Rocchio relevance feedback methods to locate and weight 50 terms and 10 statistical phrases. The statistical phrases are based on term co-occurance information for the whole collection, not just the relevant and expansion using 200 terms and syntactic phrases, with those phrases created from a full parse of the entire collection of documents. These methods can be contrasted with the INQUERY group, who started with a traditional Rocchio approach to select and weight 50 terms, but then expanded the query by 250 word pairs selected from only portions of the relevant documents.

The other three groups used less traditional methods. The group from City University repeated their very successful technique from TREC-3, in which they first used an ordering function to produce a list of terms as candidate terms for the query. This list was then optimized by repeatedly trying different sets of terms. The final term set in the *cityr2* run used an average of 36 terms per query, with the number varying across queries. The Xerox group started by expanding the query using Rocchio techniques, and used this expanded query to select 2000 documents. These 2000 documents were then fed into a LSI process to reduce the dimensionality of the final feature set. The final group, the *pircsC* run from Queens College, CUNY, was the result of four different expansions, two using different levels of expansion and two using different subcollections of documents for the expansion.

In addition to using different methods to select the features for the queries, two of the groups experimented with different ways of combining these features. The group from Xerox used three different classification techniques, combining the results from these three "experts." The *pircsC* group combined the results of their four query expansion experts. Both groups found that the combination of experts outperformed using a single method, even when one method (large expansion in the *pircs2* case and neural networks in the *xerox1* case) was generally superior. Also both groups found that there was a huge variation in performance across topics, with some topics performing best for each of the various experts.

The use of the two different subcollections of topics (25 in each set) for the routing task was, in general, not utilized by the various groups. However, it is very interesting to examine the results of the 6 groups shown in Figure 6 when broken into the two subsets. This is shown in Figure 7. The most prominant feature of these graphs is the difference in the shape of the curves. The *Federal Register* subcollection results (shown in grey) have a sharper drop in precision early in the curve, but better performance in general in the high recall end of the curve. Two

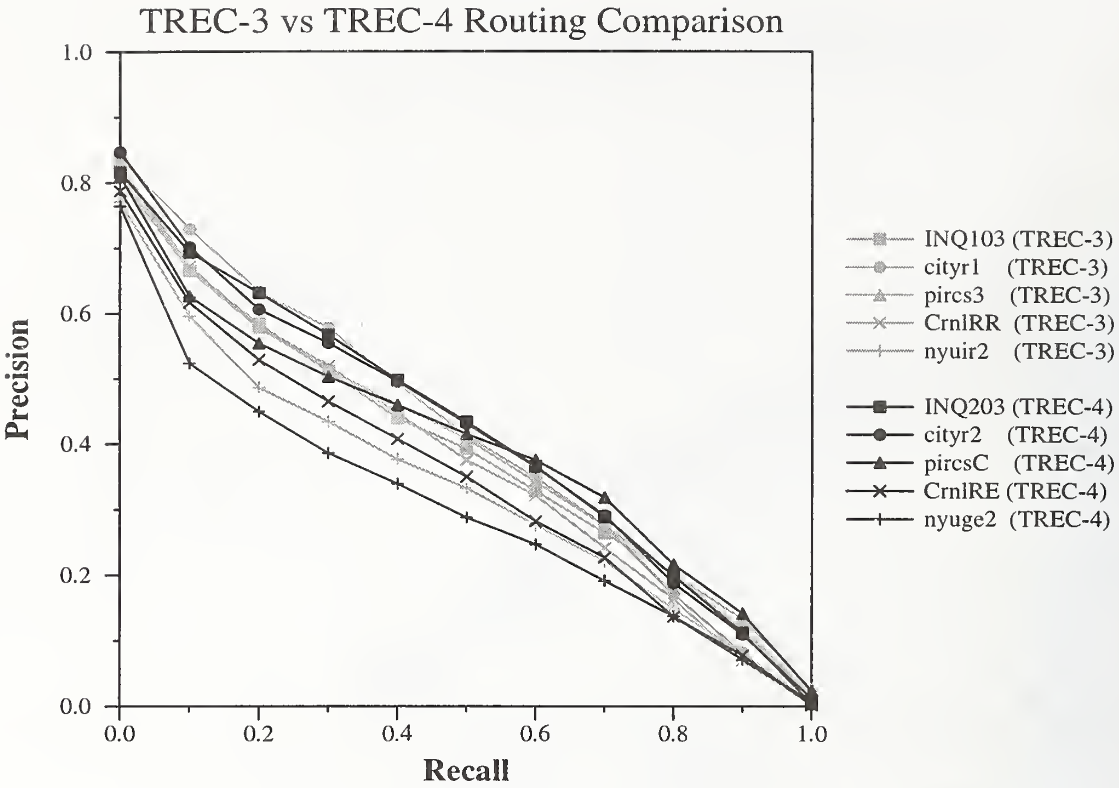


Figure 8. Comparison of Routing Results for TREC-3 and TREC-4.

differences in the subcollections account for this. First, the 25 topics in the FR subcollection retrieved significantly fewer relevant documents, an average of 99 relevant documents, as opposed to an average of 164 relevant documents for the computer topics. Additionally most of these relevant documents are *Federal Register* documents, which are very long and traditionally have been difficult to retrieve. These differences account for the sharp drop in precision in the low recall end of the curve. The higher performance of most of these 6 systems at the high recall end of the curve is somewhat more puzzling. It may be that the types of terminology in these subcollections are such that training is more effective in the FR subcollection.

Note that certain of the 6 systems seem more affected by the two subcollections. For example, the *pircsC* run is actually better for the FR subcollection than for the computer collection. This is likely because this system chunks all documents into 550 word segments, and therefore is less affected by the long FR documents. In contrast, the INQUERY system has excellent results for the computer topics, but a sharp drop in high precision results for the FR collection

There looks to be minimal improvement in overall routing results compared with those from TREC-3 (Figure 8). However, the TREC-4 topics were more difficult, particularly the FR topics. Despite the harder topics, many of the systems achieved performance improvements, especially at the high recall end of the curves. This indicates that the ability to find useful features that can retrieve the "hard-to-find" documents is growing. Such techniques as the use of word pairs from highly ranked sections of relevant documents by the INQUERY system, and the use of multiple experts in the *pircsC* and *xerox1* runs are showing promise.

6. TREC-4 TRACKS

Starting with TREC-1, there have always been groups that have pursued different goals than achieving high recall/precision performances on the adhoc and routing tasks. For example, the group from CITRI, Royal Melbourne Institute of Technology, has investigated efficiency issues in several of the TREC evaluations. By TREC-3 some of these areas had attracted several groups, all working towards the same goal. These became informal working groups, and in TREC-4 these working groups were formalized into "tracks," with specific guidelines.

6.1 The Multilingual Track

One of these tracks investigated the issues of retrieval in languages other than English. An preliminary Spanish test was run in TREC-3, with a formal track in TREC-4

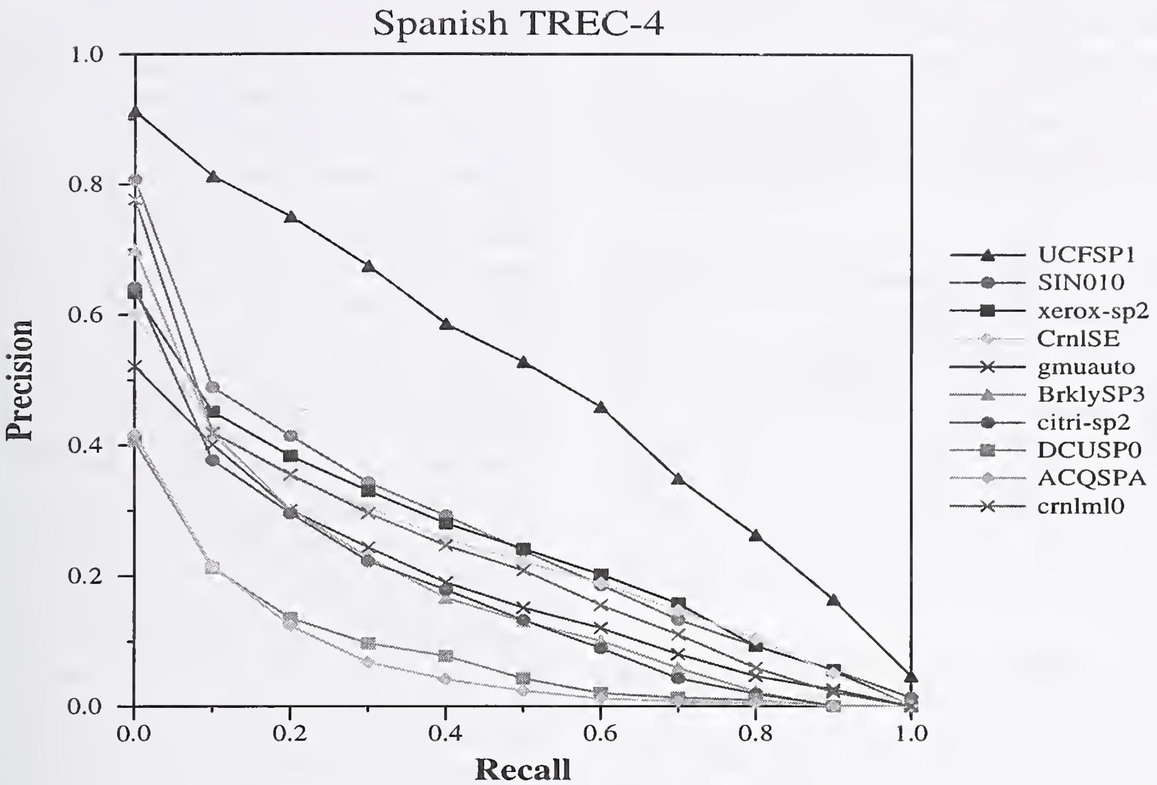


Figure 9. Results of TREC-4 Spanish Track.

that attracted 10 groups. Both TREC-3 and TREC-4 used the same documents, about 200 megabytes of the *El Norte* newspaper from Monterey, Mexico, but there were 25 different topics for each evaluation. Groups used the adhoc task guidelines, and submitted the top 1000 documents retrieved for each of the 25 Spanish topics.

The major result from TREC-3 was the ease of porting the retrieval techniques across languages. Cornell reported that only 5 hours to 6 hours of system changes were necessary (beyond creation of any stemmers or stopword lists). In TREC-4 there was training data (the results of TREC-3), and groups were able to do more elaborate testing. Figure 9 shows the recall/precision curves for these 10 TREC-4 groups, ordered by non-interpolated average precision. The cited papers are in this proceedings.

UCFSP1 -- University of Central Florida ("Multi-lingual Text Filtering Using Semantic Modeling" by James R. Driscoll, Sara Abbott, Kai-Lin Hu, Michael Miller and Gary Theis) used semantic modeling of the topics. A profile (entity-relationship schema) was manually built for each topic and lists of synonyms were constructed, including the use of an automatic Spanish verb form generator. The synonym list and domain list (instances of entities) were carefully built by Sara Abbott as part of a student summer project.

SINQ010 -- University of Massachusetts at Amherst ("Recent Experiments with INQUERY" by James Allan, Lisa Bellesteros, James P. Callan, W. Bruce Croft and Zhihong Lu) was a Spanish version of the automatic TREC-4 INQ201 run for the adhoc tests. The Spanish stemmer from TREC-3 was used, and terms were expanded using the basic InFinder technique (with a new noun phrase recognizer for Spanish).

xerox-sp2 -- Xerox Research Center ("Xerox Site Report: Four TREC-4 Tracks" by Marti Hearst, Jan Pedersen, Peter Pirolli, Hinrich Schutze, Gregory Grefenstette and David Hull) tested several Spanish language analysis tools, including a finite-state morphology and a hidden-Markov part-of-speech tagger to produce correct stemmed forms and to identify verbs and noun phrases. The SMART system was used as the basic search engine. Expansion was done using the top 20 retrieved documents.

CrnlSE -- Cornell University ("New Retrieval Approaches Using SMART: TREC-4" by Chris Buckley, Amit Singhal, Mandar Mitra, (Gerald Salton)) is a repeat of the TREC-3 work, using a simple stemmer and stopword list, and expanding by 50 terms from the top 20 documents. The TREC-3 version of SMART was used.

gmuauto -- George Mason University ("Improving Accuracy and Run-Time Performance for TREC-4" by David A. Grossman, David O. Holmes, Ophir Frieder, Matthew D. Nguyen and Christopher E. Kingsbury) used 5-grams with a vector-space type system for ranking. A Spanish stopword list was constructed using a Spanish linguist to prune a list of the most frequent 500 terms in the text.

BrklySP3 -- University of California, Berkeley ("Logistic Regression at TREC4: Probabilistic Retrieval from Full Text Document Collections" by Fredric C. Gey, Aitao Chen, Jianzhang He and Jason Meggs) trained their logistic regression method on the Spanish results from TREC-3. They also built a rule-based Spanish stemmer, including a borrowed file of all verb forms for irregular verbs. The queries were formed manually by translating them into English, searching the MELVYL NEWS database, reformulating the English queries based on these searches, and then translating the queries back into Spanish.

citri-sp2 -- RMIT, Australia ("Similarity Measures for Short Queries" by Ross Wilkinson, Justin Zobel, and Ron Sacks-Davis) tried the combination methods used for their English results. A stop-list of 316 words was created, along with a Spanish stemmer that principally removed regular verb suffixes. Experiments were done using combinations of stopped and stemmed results.

DCUSP0 -- Dublin City University ("TREC-4 Experiments at Dublin City University: Thresholding Posting Lists, Query Expansion with WordNet and POS Tagging in Spanish" by Alan F. Smeaton, Fergus Kelledy and Ruairi O'Donnell) used the NMSU part-of-speech tagger (at NMSU) as input to the SMART system. This method also produced the base forms of the terms. The traditional *tf*IDF* weighting was used, but adjectives were double-weighted.

ACQSPA -- Department of Defense ("Acquaintance: Language-Independent Document Categorization by N-Grams" by Stephen Huffman) used a 5-gram method which normalizes the resulting document vectors by subtracting a "collection" centroid vector. Minimal topic expansion was done.

crnlml0 -- New Mexico State University ("A TREC Evaluation of Query Translation Methods for Multi-Lingual Text Retrieval" by Mark Davis and Ted Dunning) investigated five different methods of query translation. The Spanish topics were first manually translated into English for use in these tests. Then five different methods were used to automatically translate the topics into Spanish. The five methods were 1) a term-by-term translation using a bilingual dictionary, 2) use of the parallel corpus (UN corpus) for high-frequency terms, 3) use of a parallel corpus to locate statistically significant terms, 4) optimization of 2) and 5) an LSI technique on the parallel corpus.

In general the groups participating in the Spanish task were using the same techniques as for English. This is consistent with the philosophy that the basic search engine techniques are language-independent. Only the auxiliary techniques, such as stopword lists and stemmers, need to be language dependent. Several of the groups did major linguistic work on these auxiliary files, such as the noun-phrase identifier necessary for expansion using InFinder (the INQUERY system) and the two new Spanish stemmers (*BrklySP3* and *citri-sp2*). Two groups used n-gram methods, as did two of the groups in TREC-3.

Several other issues unique to this track should be mentioned. First, the outstanding results from the University of Central Florida indicate the benefits of very careful building of the manual queries, in this case by building extensive synonym sets and other such lists. The utility of this technique outside the rather limited domain of the TREC-4 topic set is an open question however. The group from Xerox did extensive work with Spanish language tools, but the effort had the same type of minimal effects generally seen in English. As a final point, the query translation experiments by New Mexico State University demonstrated a very interesting approach to the problem of multilingual retrieval, and hopefully will be followed by better results in TREC-5.

This track will be run again in TREC-5, with new Spanish data and 25 new Spanish topics. Also new for TREC-5 will be a Chinese retrieval task, with Chinese data and 25 Chinese topics.

6.2 The Confusion Track

The "confusion" track represents an extension of the current tasks to deal with corrupted data such as would come from OCR or speech input. The track followed the adhoc task, but using only the category B data. This data was randomly corrupted at NIST using character deletions, substitutions, and additions to create data with a 10% and 20% error rate (i.e., 10% or 20% of the characters were affected). Note that this process is neutral in that it does not model OCR or speech input. Four groups used the baseline and 10% corruption level; only two groups tried the 20% level. Figure 10 shows the recall/precision curves for the confusion track, ordered by non-interpolated average precision. Two or three runs are shown for each group, the base run (no corruption), the 10% corruption level, and (sometimes) the 20% corruption level. The cited papers are in this proceedings.

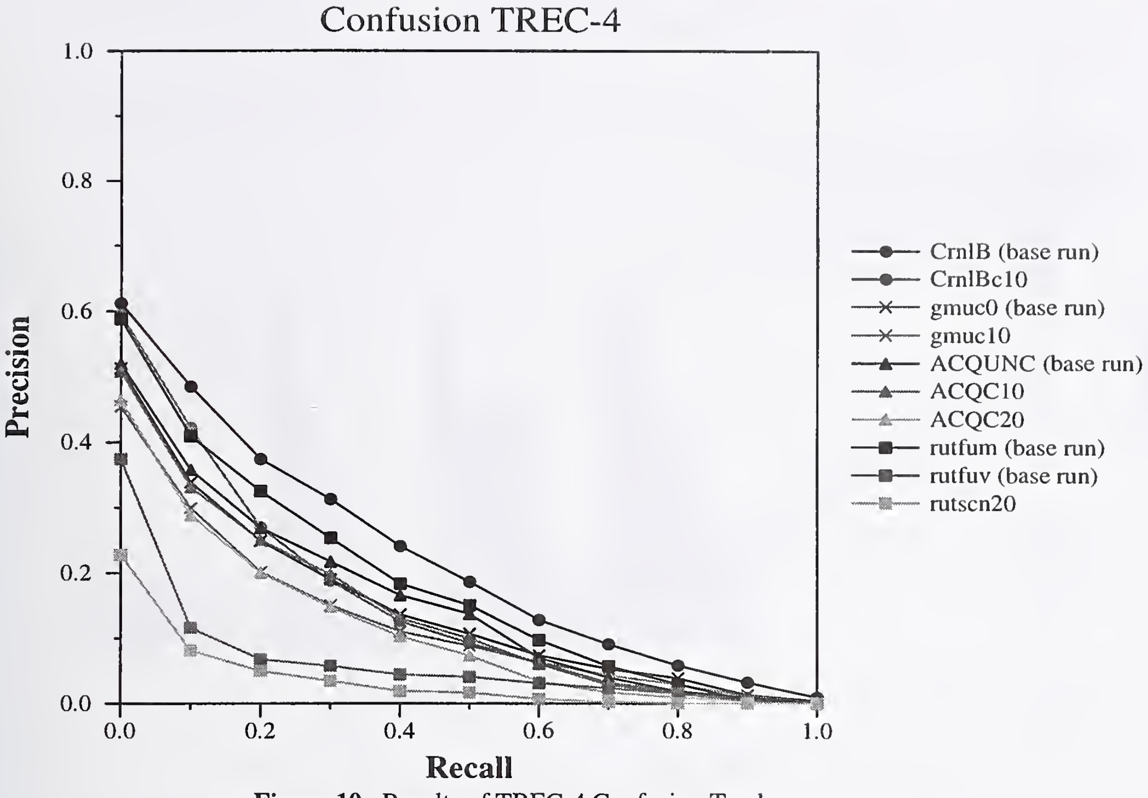


Figure 10. Results of TREC-4 Confusion Track.

CrnlB, CrnlBc10 -- Cornell University ("New Retrieval Approaches Using SMART: TREC-4" by Chris Buckley, Amit Singhal, Mandar Mitra, (Gerald Salton)) used a two-pass correction technique (only one-pass is implemented for this run). In the first pass, the query is expanded by all variants that are one transformation from the query word. The second pass improves the documents. This method avoids the use of a dictionary for correction of corrupted text.

ACQUNC, ACQC10, ACQC20 -- Department of Defense ("Acquaintance: Language-Independent Document Categorization by N-Grams" by Stephen Huffman) used an n-gram method which normalizes the resulting document vectors by subtracting a "collection" centroid vector. A 5-gram was used for the 10% corruption level and a 4-gram for the 20% level.

gmuc0, gmuc10 -- George Mason University ("Improving Accuracy and Run-Time Performance for TREC-4" by David A. Grossman, David O. Holmes, Ophir Frieder, Matthew D. Nguyen and Christopher E. Kingsbury) used a 4-gram method with a vector-space type system for ranking. A thresholding technique was tried that only worked with the best 75% of the 4-gram query in order to improve efficiency.

rutfum, rutfuv, rutscn20 -- Rutgers University ("Two Experiments on Retrieval with Corrupted Data and Clean Queries in the TREC-4 Adhoc Task Environment: Data Fusion and Pattern Scanning" by Kwong Bor Ng and Paul B. Kantor) tried the use of 5-grams and data fusion. The first experiment merged the results of two runs, one using 5-grams and one using words. The second experiment was a pattern scanning scheme called dotted 5-grams.

Since this was the first time this task had been tried, and since also there were very few participating groups, not much can be said about the results. Three of the four groups used N-grams, a method that is not known for the best results on uncorrupted data. The fourth group was unable to implement their full algorithms in time for the results. The track will be run again in TREC-5. Actual OCR output will be used at that time, as opposed to the randomly corrupted data used in TREC-4.

6.3 The Database Merging Track

A third area, that of properly handling heterogeneous collections such as the five main "subcollections" in TREC, was examined by the database merging track. This type of investigation is important for real-world collections, and also to allow researchers to take advantage of possible variations in retrieval techniques for heterogeneous collections.

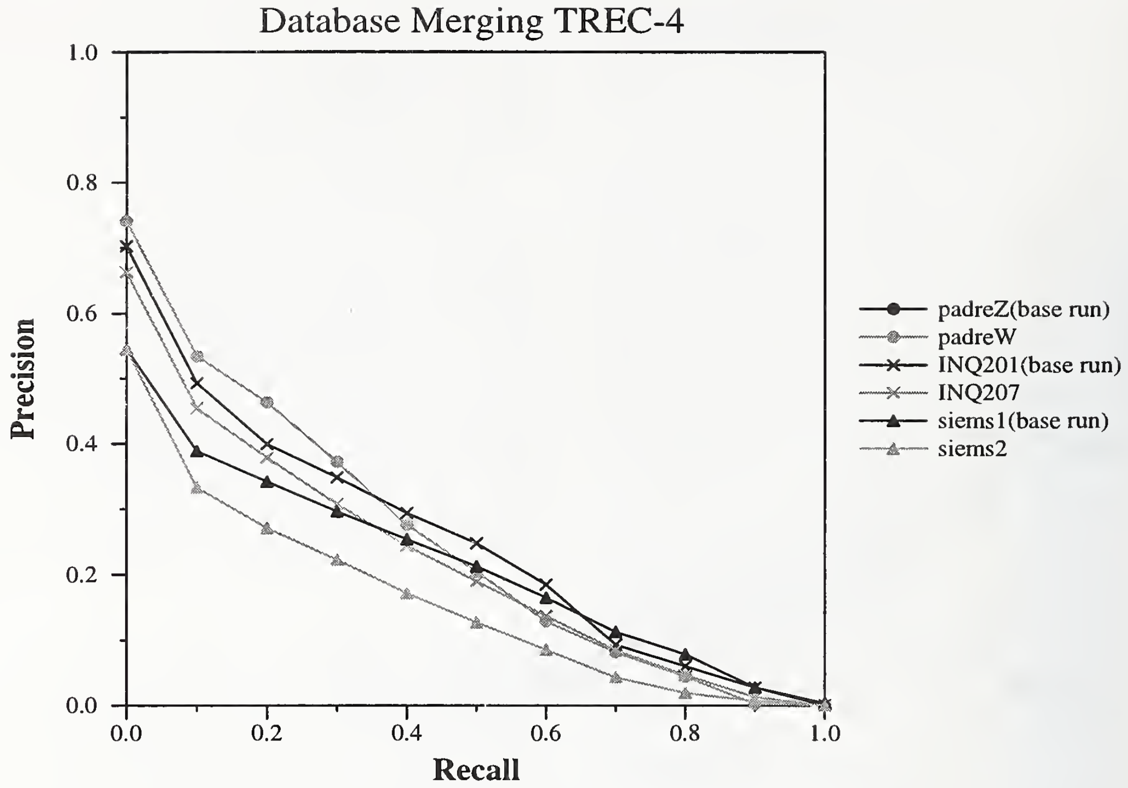


Figure 11. Results of TREC-4 Database Merging Track.

There were 10 subcollections defined corresponding to the various dates of the data, i.e., the three different years of the *Wall Street Journal*, the two different years of the *AP* newswire, the two sets of Ziff documents (one on each disk), and the three single subcollections (the *Federal Register*, the *San Jose Mercury News*, and the U.S. Patents). The 3 participating groups ran the adhoc topics separately on each of the 10 subcollections, merged the results, and submitted these results, along with a baseline run treating the subcollections as a single collection.

Figure 11 shows the recall/precision curves for this track, ordered by non-interpolated average precision. Two runs are shown for each group, the base run (indexed as a single database), and the best of their merged runs. The cited papers are in this proceedings.

padreZ, padreW -- Australian National University ("Proximity Operators -- So Near and Yet So Far" by David Hawking and Paul Thistlewaite) used manual queries with proximity operators. Since there are no collection-dependent variables in this system, the run using the 10 separate collections is equivalent to the run using the entire collection.

INQ201, INQ207 -- University of Massachusetts at Amherst ("Recent Experiments with INQUERY" by James Allan, Lisa Bellesteros, James P. Callan, W. Bruce Croft and Zhihong Lu) tried five variations of a basic method of collection merging [Callan et al. 1996]. The basic method scored each collection against the topic, and then weighted the document results by their collection score.

siems1, siems2 -- Siemens Corporate Research ("Siemens TREC-4 Report: Further Experiments with Database Merging" by Ellen M. Voorhees) tried two different methods, both based on information about the previous queries (training topics) as opposed to using information about the document collection itself.

If results are produced without use of collection information, then the merging process is trivial, as illustrated by the *padre* runs. Certainly this is one method of handling the problems of merging results from different databases. However this precludes using information about the collection to modify the various algorithms in the search engine, and, even more importantly, it does not deal with the issue about which collection to select. An implied question in this track is the hypothesis that one might want to bias searching towards certain collections, either by developing collection scores (such as the INQUERY work) or by developing a sense of history from previous queries (the Siemens work).

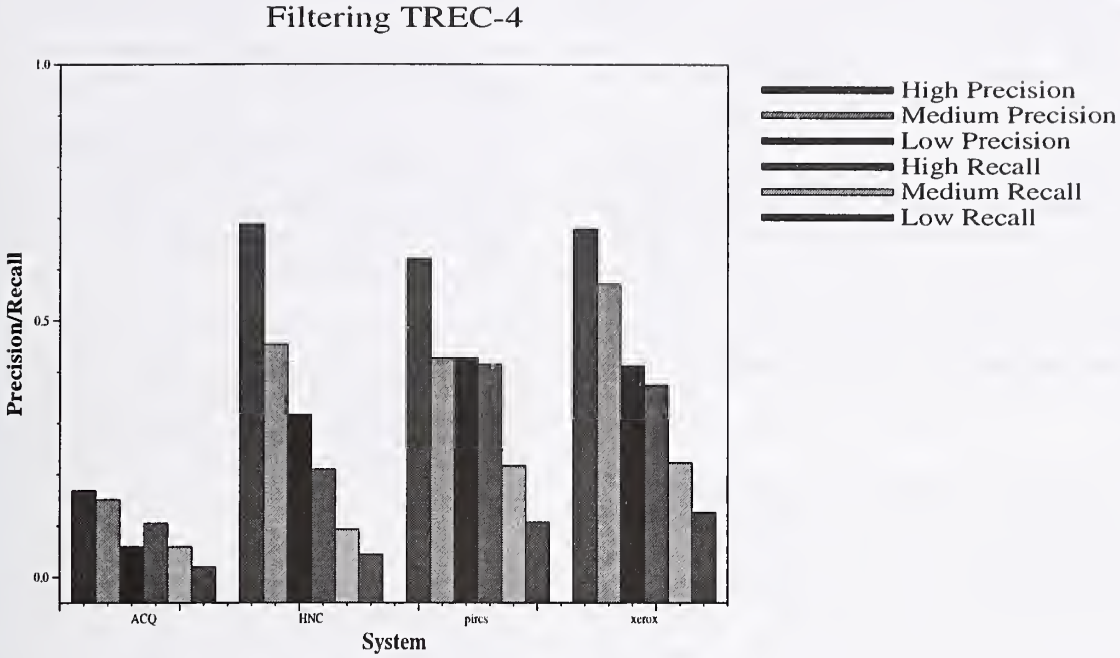


Figure 12. Results of TREC-4 Filtering Track.

6.4 The Filtering Track

For several years some participants have been concerned about the definition of the routing task, and the filtering track represents a new variation of this task. In TREC-4 this track documents, and test documents as the routing task. The difference was that the results submitted for the filtering runs were unranked sets of documents satisfying three "utility function" criteria. These criteria were designed to approximate a high precision run, a high recall run, and a "balanced" run. For more details, see the paper "The TREC-4 Filtering Track" by David Lewis (in this proceedings).

Figure 12 shows the results of the four groups that tried this track. There are 3 pairs of bars for each system, one pair corresponding to each of the three utility function criteria. The first of the pairs (the left-most and the right-most bars) correspond to the high precision/low recall run. The second pair (the second and fifth bars) correspond to the balanced (medium precision/medium recall) run, and the third pair (high recall/low precision run) are shown in the middle two bars.

One desired type of system behavior is the "stairstep" effect seen, for example, in the run from HNC Software Inc. (see paper "Using CONVECTIS, A Context Vector-Based Indexing System for TREC-4" by Joel L. Carleton, William R. Caid and Robert V. Sasseen in this proceedings). When this system is compared with the next two systems (*pircs* and *xerox*) , it can be seen that while the HNC system got a better separation of the runs, the other two groups got better results in general, particularly for the balanced run.

This was the first time this track had been tried, and the development of evaluation techniques was the most critical area. Now that these techniques are in place, it is expected that more groups will take part in the track in TREC-5.

6.5 The Interactive Track

An interactive track was formed for TREC-4, with the double goal of developing better methodologies for interactive evaluation and investigating in depth how users search the TREC topics. Eleven groups took part in this track in TREC-4, using a subset of the adhoc topics. Many different types of experiments were run, but the common thread was that all groups used the same topics, performed the same task(s), and recorded the same information about how the searches were done. Task 1 was to retrieve as many relevant documents as possible within a certain timeframe. Task 2 was to construct the best query possible. The cited papers are in this proceedings.

rutint1, *rutint2* -- Rutgers University ("Using Relevance

Feedback and Ranking in Interactive Searching" by Nicholas J. Belkin, Colleen Cool, Jurgen Koenemann, Kwong Bor Ng and Soyeon Park) recruited 50 searchers for this task. The INQUERY search engine was used, and the particular emphasis was on studying the use of ranking and relevance feedback by these searchers.

cityi1 -- City University, London ("Okapi at TREC-4" by S.E. Robertson, S. Walker, S. Jones, M.M. Hancock-Beaulieu and M. Gatford) used members of their team to evaluate their new GUI interface to OKAPI. They concentrated on examining the various stages of searching, and kept notes on items of interest, such as how many titles were examined, how many iterations were run, and how the queries were edited at various times in the search process.

UofTo1 -- University of Toronto ("Is Recall Relevant? An Analysis of How User Interface Conditions affect Strategies and Performance in Large Scale Text Retrieval" by Nipon Charoenkitkarn, Mark H. Chignell and Gene Golovchinsky) used 36 searchers on a new version of their system called BrowsIR. The goal of their experiments was to compare three different strategies for constructing queries: a text markup (similar to that done by this group in TREC-3), a query typing method, and a hybrid method. Both experts and novices were used.

ETHI01 -- Swiss Federal Institute of Technology (ETH) ("Highlighting Relevant Passages for Users of Interactive SPIDER Retrieval System" by Daniel Knaus, Elke Mittendorf and Peter Schäuble and Paraic Sheridan) experimented with several algorithms to highlight the most relevant passages, and tested this on 11 users as an aid to relevance feedback.

XERINT1, XEROXINT2 -- Xerox Research Center ("Xerox Site Report: Four TREC-4 Tracks" by Marti Hearst, Jan Pedersen, Peter Pirolli, Hinrich Schutze, Gregory Grefenstette and David Hull) tried three different modes of searching interfaces. The first was the Scatter/Gather method of visualizing the document space, the second was the TileBars to visualize the documents, and the third was the more traditional ranked list of titles from a vector space search engine.

CLARTI -- CLARITECH Corporation ("CLARIT TREC-4 Interactive Experiments" by Natasa Milic-Frayling, Cheng-Xiang Zhai, Xiang Tong, Michael P. Mastroianni, David A. Evans and Robert G. Lefferts) used the CLARIT system interactively to study the effects of the quality of a user's relevance judgments, the effects of time constraints on searching, and the effects of relevance feedback on the final results of queries.

LNBOOL -- Lexis-Nexis ("Interactive Boolean Search in TREC4" by David James Miller, John D. Hold and X. Allan Lu) used expert Boolean searchers and the commercial Lexis-Nexis software to compare retrieval performance between Boolean and non-Boolean systems.

gatin1, gatin2 -- Georgia Institute of Technology ("Interactive TREC-4 at Georgia Tech" by Aravindan Veerasamy) investigated the effectiveness of a new visualization tool that shows the distribution of query terms across the document space.

ACQINT -- Department of Defense ("Acquaintance: Language-Independent Document Categorization by N-Grams" by Stephen Huffman) used the Parentage information visualization system which shows clusters of documents, along with the terms which characterize those clusters.

CrnlI1, CrnlI2 -- Cornell University ("New Retrieval Approaches Using SMART: TREC-4" by Chris Buckley, Amit Singhal, Mandar Mitra, (Gerald Salton)) did an experiment to test how much of the document needed to be read in order to determine document relevancy for input to relevance feedback. They tested quick scans vs full reading.

Whereas all participants found the track very interesting and useful, there were difficulties in comparing results. One of the major outcomes of this track in TREC-4, therefore, was a general awareness of the large number of variables that need to be controlled in order to compare results. Some of these, such as the variation in performance across topics, affect all the TREC tasks, but the human element in the interactive track compounds the problem immensely. The emphasis for TREC-5 work will be on learning to control or monitor some of these variables as a first step to providing better evaluation methodology.

7. Summary

The main conclusions that can be drawn from TREC-4 are as follows:

- The much shorter topics in the adhoc task caused all systems trouble. The expansion methods used in TREC-3 continued to work, but obviously needed modifications. The types of passage retrieval used in TREC-3 did not work. The fact that the performance of the manually built queries was also hurt by the short topics implies that there are some issues involving the use of very short topics in TREC that need further investigation. It may be that the statistical "clues" presented by these shorter topics are simply not enough to provide good retrieval performance in the batch testing environment of TREC. The topics to be used in TREC-5 will contain both a short and a long version to aid in these further investigations.

- Despite the problems with the short topics, many of the systems made major modifications to their term weighting algorithms. In particular, the SMART group from Cornell University and the INQUERY group from the University of Massachusetts at Amherst produced new algorithms that yielded much better results (on the longer TREC-3 queries), and their TREC-4 results were not lowered as much as they would have been.
- There were five tracks run in TREC-4.
  - Interactive -- 11 groups investigated searching as an interactive task by examining the process as well as the outcome. The major result of this track, in addition to interesting experiments, was an awareness of the difficulties of comparing results in an interactive testing environment.
  - Multilingual -- 10 groups working with 250 megabytes of Spanish and 25 topics verified the ease of porting to a new language (at least in a language with no problems in locating word boundaries). Additionally some improved Spanish stemmers were built.
  - Multiple database merging -- 3 groups investigated techniques for merging results from the various TREC subcollections.
  - Data corruption -- 4 groups examined the effects of corrupted data (such as would come from an OCR environment) by using corrupted versions of the category B TREC data.
  - Filtering -- 4 groups evaluated routing systems on the basis of retrieving an unranked set of documents optimizing a specific effectiveness measure.

  The results from these last 3 tracks were inconclusive, and should be viewed as a first-pass at these focussed tasks.

  There will be a fifth TREC conference in 1996, and most of the systems that participated in TREC-4 will be back, along with additional groups. The routing and adhoc tasks will be done again, with different data, and new topics similar in length to the TREC-3 topics. In addition, all five tracks will be run again, with new data. The Multilingual track will be run with Spanish and, as a first time, with Chinese data and topics.

Acknowledgments

The author would like to gratefully acknowledge the continued support of the Intelligent Systems Office of the Defense Advanced Research Projects Agency for the TREC conferences. Special thanks also go to the TREC program committee and the staff at NIST.

7. REFERENCES

Callan J.P., Lu Z., and Croft W.B. (1996). Searching Distributed Collections with Inference Networks. In: *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 21-29.

Harman D. (Ed.). (1994). *Overview of the Third Text REtrieval Conference (TREC-3)*. National Institute of Standards and Technology Special Publication 500-225, Gaithersburg, Md. 20899.

Sparck Jones K. and Van Rijsbergen C. (1975). *Report on the Need for and Provision of an "Ideal" Information Retrieval Test Collection*, British Library Research and Development Report 5266, Computer Laboratory, University of Cambridge.

New Retrieval Approaches Using SMART : TREC 4

Chris Buckley[*] Amit Singhal, Mandar Mitra, (Gerard Salton)

Abstract

The Smart information retrieval project emphasizes completely automatic approaches to the understanding and retrieval of large quantities of text. We continue our work in TREC 4, performing runs in the routing, ad-hoc, confused text, interactive, and foreign language environments.

Introduction

For over 30 years, the Smart project at Cornell University has been interested in the analysis, search, and retrieval of heterogeneous text databases, where the vocabulary is allowed to vary widely, and the subject matter is unrestricted. Such databases may include newspaper articles, newswire dispatches, textbooks, dictionaries, encyclopedias, manuals, magazine articles, and so on. The usual text analysis and text indexing approaches that are based on the use of thesauruses and other vocabulary control devices are difficult to apply in unrestricted text environments, because the word meanings are not stable in such circumstances and the interpretation varies depending on context. The applicability of more complex text analysis systems that are based on the construction of knowledge bases covering the detailed structure of particular subject areas, together with inference rules designed to derive relationships between the relevant concepts, is even more questionable in such cases. Complete theories of knowledge representation do not exist, and it is unclear what concepts, concept relationships, and inference rules may be needed to understand particular texts.[8]

Accordingly, a text analysis and retrieval component must necessarily be based primarily on a study of the available texts themselves. Fortunately very large text databases are now available in machine-readable form, and a substantial amount of information is automatically derivable about the occurrence properties of words and expressions in natural-language texts, and about the contexts in which the words are used. This information can help in determining whether a query and a text are semantically homogeneous, that is, whether they cover similar subject areas. When that is the case, the text can be retrieved in response to the query.

Automatic Indexing

In the Smart system, the vector-processing model of retrieval is used to transform both the available information requests as well as the stored documents into vectors of the form:

$$D_i = (w_{i1}, w_{i2}, \ldots, w_{it})$$

where D_i represents a document (or query) text and w_{ik} is the weight of term T_k in document D_i. A weight of zero is used for terms that are absent from a particular document, and positive weights characterize terms actually assigned. The assumption is that t terms in all are available for the representation of the information.

In choosing a term weighting system, low weights should be assigned to high-frequency terms that occur in many documents of a collection, and high weights to terms that are important in particular documents but unimportant in the remainder of the collection. The weight of terms that occur rarely in a collection is

[*]Department of Computer Science, Cornell University, Ithaca, NY 14853-7501. This study was supported in part by the National Science Foundation under grant IRI 93-00124.

relatively unimportant, because such terms contribute little to the needed similarity computation between different texts.

A well-known term weighting system following that prescription assigns weight w_{ik} to term T_k in query Q_i in proportion to the frequency of occurrence of the term in Q_i, and in inverse proportion to 'the number of documents to which the term is assigned.[9, 7] Such a weighting system is known as a tf×idf (term frequency times inverse document frequency) weighting system. In practice the query lengths, and hence the number of non-zero term weights assigned to a query, varies widely.

The terms T_k included in a given vector can in principle represent any entities assigned to a document for content identification. In the Smart context, such terms are derived by a text transformation of the following kind:[7]

1. recognize individual text words
2. use a stop list to eliminate unwanted function words
3. perform suffix removal to generate word stems
4. optionally use term grouping methods based on statistical word co-occurrence or word adjacency computations to form term phrases (alternatively syntactic analysis computations can be used)
5. assign term weights to all remaining word stems and/or phrase stems to form the term vector for all information items.

Once term vectors are available for all information items, all subsequent processing is based on term vector manipulations.

The fact that the indexing of both documents and queries is completely automatic means that the results obtained are reasonably collection independent and should be valid across a wide range of collections. No human expertise in the subject matter is required for either the initial collection creation, or the actual query formulation.

Phrases

The same phrase strategy (and phrases) used in all previous TRECs ([4, 2, 5]) are used for TREC 4. Any pair of adjacent non-stopwords is regarded as a potential phrase. The final list of phrases is composed of those pairs of words occurring in 25 or more documents of the initial TREC 1 document set. Phrase weighting is again a hybrid scheme where phrases are weighted with the same scheme as single terms, except that normalization of the entire vector is done by dividing by the length of the single term sub-vector only. In this way, the similarity contribution of the single terms is independent of the quantity or quality of the phrases.

Text Similarity Computation

When the text of document D_i is represented by a vectors of the form $(d_{i1}, d_{i2}, \ldots, d_{it})$ and query Q_j by the vector $(q_{j1}, q_{j2}, \ldots, q_{jt})$, a similarity (S) computation between the two items can conveniently be obtained as the inner product between corresponding weighted term vector as follows:

$$S(D_i, Q_j) = \sum_{k=1}^{t} (d_{ik} * q_{jk}) \tag{1}$$

Thus, the similarity between two texts (whether query or document) depends on the weights of coinciding terms in the two vectors.

System Description

The Cornell TREC experiments use the SMART Information Retrieval System, Version 12, and are run on a dedicated Sun Sparc 20/51 with 160 Megabytes of memory and 27 Gigabytes of local disk.

SMART Version 12 is the latest in a long line of experimental information retrieval systems, dating back over 30 years, developed under the guidance of G. Salton. The new version is approximately 44,000 lines of C code and documentation.

SMART Version 12 offers a basic framework for investigations of the vector space and related models of information retrieval. Documents are fully automatically indexed, with each document representation being a weighted vector of concepts, the weight indicating the importance of a concept to that particular document (as described above). The document representatives are stored on disk as an inverted file. Natural language queries undergo the same indexing process. The query representative vector is then compared with the indexed document representatives to arrive at a similarity (equation (1)), and the documents are then fully ranked by similarity.

Document length normalization

It has become increasingly obvious over the past two years that the standard SMART method of document length normalization, cosine normalization, does not work optimally in the TREC environment.

The cosine similarity function (or equivalently, cosine document normalization) was developed in an era in which documents were short, and about a single topic. It emphasizes the relationship between the query and the entire document. Negative information, the fact that large parts of the document are *not* related to the query, is just as important as positive information.

Use of negative information is no longer appropriate if there are longer, full text documents to be retrieved. These long documents will have several sub-topics, only one of which may be pertinent to a query. Normalization using the cosine function will make these documents very difficult to retrieve, since the negative information will dominate.

Figure 1 shows the mismatch between the probability of retrieval of cosine normalized documents of a given length, and the probability of relevance of documents of that length. In an ideal graph where the system was accurately retrieving documents independent of length effects, these two curves would be co-incident.

The documents used for the TREC 4 ad hoc task were sorted by length into 568 bins of 1000 documents each. For each TREC 4 query we retrieve 1000 documents using our standard cosine normalized "lnc.ltc" weighting function and analyze which buckets the retrieved documents occur in, and which buckets the relevant documents occur in. I.e. for each Bin_i, we calculate Prob(Bin_i — Relevant) and Prob(Bin_i — Retrieved) Those numbers are plotted as the y-axis of Graph 1, with the x-axis being the median length of the documents within the sample buckets.

As would be expected, the probability of relevance of a document increases with length. Longer documents have more of a chance of having a relevant sub-topic since they have more sub-topics. However, the probability of retrieval of our cosine normalized documents does not at all match this increase! In fact the probability of retrieval remains roughly constant up until a length of 3000 bytes and then starts decreasing. Thus, our "lnc.ltc" measure retrieves a much larger share of short documents than it should, and a much smaller number of long documents.

This bias towards short documents affects much more than just straightforward adhoc retrieval. Our work in massive query expansion and our local/global approach have been strongly influenced by the bias. Over half of the effectiveness increases of each of these two approaches are due to their indirectly overcoming this bias, and being able to retrieve long documents.

Our local/global matching has given us 15% improvement in past tests [5]. The local match on fixed size windows has been explicitly non-normalized. Thus short and long documents have been treated equally on the local match, making up for the biased global match. If a good, non-biased normalization approach is used for the global match, then the improvement due to our current local match is reduced to about 3%.

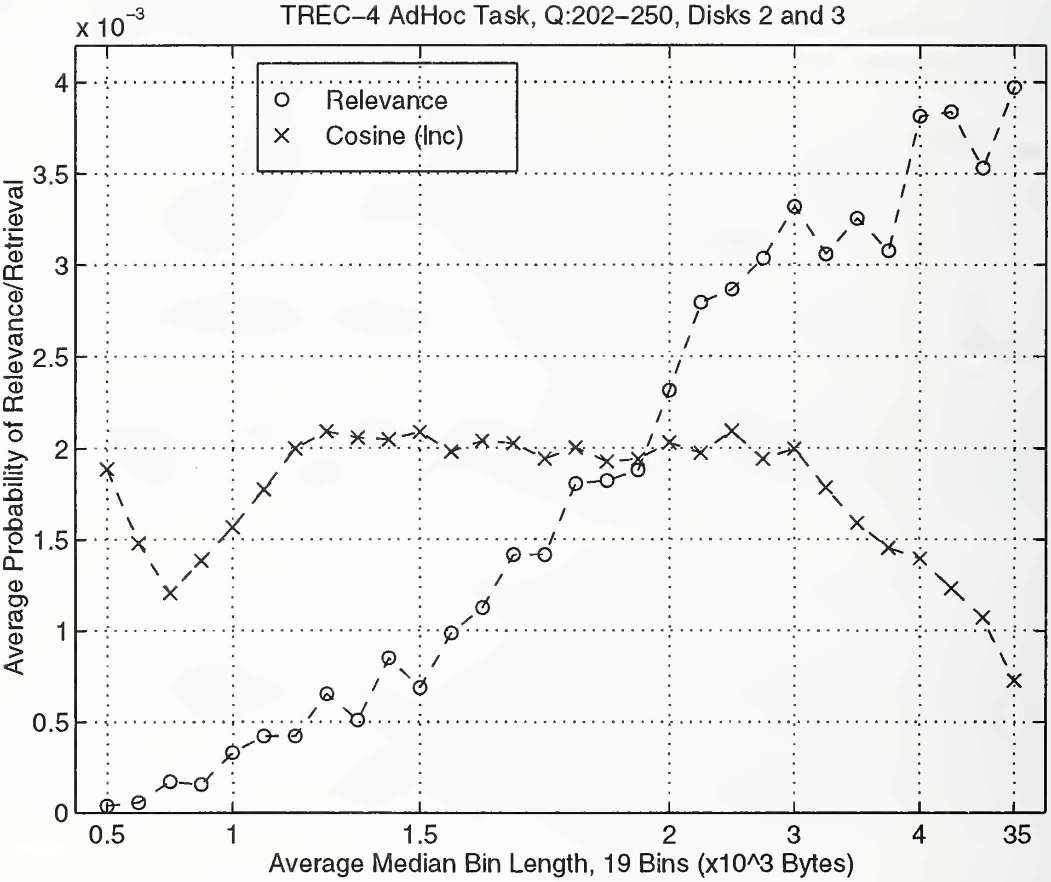


Figure 1: Probability of relevance and cosine retrieval for varying document length.

The effect on massive query expansion is a bit more subtle and harder to isolate. Consider the effect of adding 100 random (not related to relevance) common terms to a query. This will not have a large effect on a given short document since those terms are not likely to occur. A long document will be much more strongly affected since these terms will occur by "random chance". Thus long documents will have a higher comparative similarity than short documents due to these added terms, and we have an effect counter-balancing the short document bias of the cosine normalization. The end result has been an effective similarity approach, achieved through combining two biased approaches.

In our past work,[2, 5, 1], we've been expanding by 200 – 300 terms before reaching a point of diminishing returns. In contrast, starting with a non-biased normalization the maximum effect occurs at 80 – 100 added terms, and there is a slight decrease of 2% if 300 terms are added. For most of our test sets, the end result of the expansion is 10% – 15% better with the non-biased normalization; the expansion terms themselves do not help as much in the non-biased case, but the base non-expanded similarity starts off much better.

The background and derivation of the weighting function Cornell used in TREC 4 is described in Singhal et al [11]. Normalization is based upon both normalizing the *tf* factor by the average *tf* in the vector, and the overall vector length by a factor dependent on the number of unique terms. Based on this *tf* factor (which we call the *L* factor in Smart's term weight triple notation [9]) and pivoted unique normalization (which we call the *u* normalization), we obtain the final weighting strategy of the documents (called *Lnu* weighting in Smart):

$$\frac{\frac{1+log(tf)}{1+log(average\ tf)}}{(1.0 - slope) \times pivot + slope \times \#\ of\ unique\ terms}$$

Within our experiments here, *slope* is fixed at 0.2, and the pivot is set to the average number of unique terms occurring in the collection.

A major portion of our work since TREC 3 has been concerned with document length normalization [11], and the ramifications. The measure presented here will not be our final normalization measure, but it performs equivalently. We want the final measure to be independent of stemming and erroneous text like misspellings and OCR errors.

CrnlAE – Adhoc expansion run

The first of Cornell's two ad-hoc runs, CrnlAE, is very similar to our TREC 3 CrnlAE run. Last year, an initial retrieval was done, the top 30 documents were assumed to be relevant and submitted to our standard Rocchio relevance feedback procedure. The query was then expanded by 500 terms and 10 phrases and resubmitted for retrieval (without the user ever having seen the first retrieval's results).

The differences between the details of that approach and this year's TREC 4 approach are due to two factors: the new document length normalization approach, and the much shorter queries. The normalization approach implies the number of expansion terms should be cut from 500 to, say, 100. The shorter queries imply that it may be easier to lose the focus of the expanded query on the original topic. So the expansion amount is reduced even further, to 50 single terms and 10 phrases, and the number of documents that the expansion is based upon is reduced to 20. The validity of this last assumption and reduction needs to be tested; but this hasn't been done yet.

Thus, the TREC 4 CrnlAE procedure is as follows: The new *Lnu* weighting scheme is used for the documents, with *ltu* weights for the original queries. These queries are initially run against the documents, retrieving the top 20 documents. These documents are marked relevant without examining them, and they and the original query are fed into the Rocchio relevance feedback process. The most frequently occurring 50 single terms and 10 phrases from the top 20 documents are added to query, and the new vector is weighted by

$$\begin{aligned} Q_{\text{new}} &= A * Q_{\text{old}} \\ &+ B * \text{average\_wt\_in\_rel\_docs} \\ &- C * \text{average\_wt\_nonrel\_docs} \end{aligned}$$

Run	Rec-Prec	Relevant-ret
lnc.ltc	1627 (-)	3210
lnc.ltc-Exp	2012 (+24%)	3634
Lnu.ltu	2326 (+43%)	3709
Lnu.ltu-Exp	2944 (+81%)	4350

Table 1: Comparisons of adhoc normalization and expansion

The *A.B.C* parameters of the Rocchio equation are set to 8.8.0. These parameters weight the original query terms higher than in standard relevance feedback, and disregard occurrences among the non-relevant documents.

The new query is re-run against the *Lnu* weighted documents, retrieving the final 1000 documents submitted as the official CrnlAE run.

Table 1 shows the effect of good normalization on the TREC 4 adhoc task. This is a very large increase for both the unexpanded and expanded cases. The increase is much larger than the 20% that had been observed in the development of the *Lnu* weighting scheme. We conjecture this is due to the much shorter queries of TREC 4, which are particularly unsuited for a cosine similarity function.

Our results compared to the other TREC 4 adhoc systems look very good, especially considering that the CrnlAE run is completely automatic, and a number of the other good runs involved substantial human involvement. We are best on only 2 of the 49 queries, but are above the median on 41 queries.

CrnlAL – Adhoc individual term locality run

Continuing in Cornell's tradition of presenting one development run, for which we have evidence it will work, and one experimental run, for which we hope it might work, we develop an entirely new similarity function, Individual Term Locality (ITL for short).

The overall ITL approach is like that of Cornell's local/global approaches of the past few years. An initial retrieval is done using global criteria, and the top retrieved documents are re-indexed and re-ranked using criteria based on matches within a certain locality or part of the documents. The difference is that using ITL, no attempt is made to break a document down into its component parts. Instead the entire document is repesented by a sequence of tuples such that for every location in the document for which there is a query term match, the term, location, and possible other information is kept. This is not a vector and if a query term occurs more than once in the document, it will occur more than once in the locality tuple list.

For every point in the document, we then estimate the point similarity of the text around that specific point to the query. The total ITL similarity of the document to the query is the maximum of the point similarities for all points within the document.

The point similarity is calculated by sorting the locality tuple list by increasing distance of each tuple from the point under consideration. The sorted list is gone through in increasing distance order, summing the contributions of the individual tuples. The tuple contribution is the product of several contributions, based upon the following factors:

1. Distance of the tuple from the point. The greater the distance, the less this tuple should contribute to whether the document is relevant around that point.

2. Number of times this tuple term has been seen before in this point similarity computation. A term occurring a second time shouldn't contribute as much as it did the first time it occurred (within this sorted distance tuple list).

3. Weight of the term in the query.

4. Certainty of term match. This is the constant 1.0 within the adhoc task, but in an erroneous environment such as OCR or speech retrieval, would be affected by the probability of this being a correct match.

Run	Best	≥ median	< median
CrnlAE	2	39	8
CrnlAL	3	36	10

Table 2: Comparative Ad-hoc results

Run	Recall- Precision	Total Rel Retrieved	R Precision	Precision 100 docs
CrnlAE	2944	4350	3384	3112
CrnlAL	2829	4297	3256	3002

Table 3: Ad-hoc results

5. Overall document length. A document length normalization effect that in all of our experiments so far can be completely ignored.

6. Relationship of this term to its immediately surrounding terms. A lot of different things can go into this factor. Some are

   (a) Whether surrounding terms occur closely together in the query.

   (b) Whether surrounding terms occur closely together in a set of relevant documents.

   (c) Syntactic relationship between terms (same noun phrase?)

   (d) Semantic relationship between terms

   For our experiments in the adhoc and routing tasks, we only considered the first two relationships (for the adhoc task, we worked in an initial retrieval environment, with the top documents being considered relevant for the purposes of the second relationship).

The benefit of ITL is that individual term occurrences are being considered. This is not a big benefit for ordinary statistical retrieval; Cornell and others have been looking at statistical local and passage matching for at least the past 8 years. However, it offers great potential for establishing a framework to bring in non-statistical information into a similarity computation. In particular, the certainty and linguistic factors above can have an influence at the individual term level. Thus ITL should be very useful for NLP, OCR, and retrieval of speech.

The present drawback of ITL is that it has not been established that the pure statistical factors of ITL can be used to rank documents as effectively as the normal global similarities. Once that can be shown, then incorporating the additional non-statistical information into the equation should improve retrieval.

The CrnlAL official adhoc run is a 3 pass run. The CrnlAE run is duplicated as the first two passes. The end result of the two passes for each query is an expanded query, a list of 20 presumed relevant documents, and a ranked list of 1750 documents. The presumed relevant documents are analyzed to determine the degree to which pairs of expanded query terms are statistically related to each other. Then each of the 1750 top documents are re-indexed relative to the expanded query, getting the occurrence location of each query term match. The ITL similarity function is calculated for each document by calculating the point similarity at each term match within the document, and taking the maximum point similarity over all term matches. The final similarity of the document was set to be the global similarity plus the ITL similarity.

The results are very good. CrnlAE was surprisingly very consistently above the median considering the short queries, and the possibility of losing the focus of the query if the initial retrieval did not perform well. We had expected the very good results on many queries to be partly counter-balanced by very poor results on others. The results for CrnlAL were also good, resulting in more "best" scores, but were less consistent.

The CrnlAL results are very encouraging, but for now say that the ITL approach using only statistical information is not quite as good as using the purely global statistical approach. The official average precision figure of .2829 is about 4% worse than the CrnlAE run. This agrees well with our preliminary tests on other TREC subcollections in which ITL by itself was 8% worse than the global similarity. This difference is noticable but not large. Given the fact that our global run improved anywhere from 12% to 45% this past year (depending on the measure), the ITL approach appears valid and ready for incorporation of non-statistical information.

Routing Experiments

The basis for our development routing experiment, CrnlRE, in TREC 4 is the same as in TREC 3, the relevance feedback approach of Rocchio [6, 10, 2]. The TREC 4 CrnlRE run starts with "Lnu" weighted documents and "ltu" weighted queries. Expressed in vector space terms, the final query vector is this initial query vector moved toward the centroid of the relevant documents, and away from the centroid of the non-relevant documents.

$$\begin{aligned} Q_{\text{new}} &= A * Q_{\text{old}} \\ &+ B * \text{average\_wt\_in\_rel\_docs} \\ &- C * \text{average\_wt\_nonrel\_docs} \end{aligned}$$

Terms that end up with negative weights are dropped (less than 3% of terms were dropped in the most massive query expansion below).

The CrnlRE run uses 64,64,2 as the A,B,C parameters in the above equation, and expands the query by adding the 50 single terms and 10 phrases that occur in the most relevant documents. This emphasizes the original query terms a bit more than in past TREC's and and reduces the expansion amount significantly. Experiments suggest that this change is an over-reaction to the new weighting approaches, and that more expansion terms weighted more heavily should be used. More details of these experiments will be given in the final version of this paper.

Unlike in TREC 3, after forming the feedback query, CrnlRE adds a separate stage of tweaking the query term weights even further on a per query basis. Dynamic Feedback Optimization, described in detail in SIGIR '95 [3], alters the weights by testing whether a mildly changed term weight performs better when run on the learning set of documents (those documents already seen and judged). If the changed weight performs better, then the new weight is kept; otherwise the weight reverts back to what it was originally. The six-pass DFO algorithm described in the SIGIR paper is run, with the DFO parameters as described in that paper. An individual term weight might be increased by about a factor of 5 over its original feedback weight.

The DFO modified CrnlRE queries are then ready to be run against the test set documents. The test set documents are weighted with the same "Lnu" scheme as used throughout this paper. The "Lnu" weights are independent of test collection statistics; the constant values for slope and pivot used in document normalization are the same as was used for the learning set. The final similarity is a simple inner product of the query and document weights, retrieving the top 1000 documents to be evaluated by NIST.

The second of the Cornell routing runs, CrnlRL, is a counterpart to the CrnlAL adhoc run. The same Rocchio procedure for forming a new feedback query is performed, except that only 10 terms and 2 phrases are added from the relevant documents. The DFO procedure is not performed for CrnlRL; the feedback query is ready to be run on the test set of documents immediately after the Rocchio procedure.

However, unlike CrnlRE, the actual running of the CrnlRL queries on the test documents is a two-pass procedure. The first pass calculates a global similarity using an inner product on the "Lnu" weighted documents. Then the second pass uses the ITL (Individual Term Locality) algorithm exactly as is used for the CrnlAL run. As well as attempting to increase precision by using a local match, it is to be hoped that all the information about tight clusters of coocurring terms derived from the learning set relevant documents can be taken advantage of.

Routing Results

Both CrnlRE and CrnlRL do reasonably but not spectacularly in comparison with other TREC 4 routing runs (Table 4). Average precision is above the median for the majority of the queries for both runs. (Table 5

Run	Best	≥ median	< median
CrnlRE	2	33	15
CrnlRL	0	30	20

Table 4: Comparative Routing Results

	Run	*X.Y*	*A.B.C*	R-prec	Total Rel	recall-prec
1.	CrnlRE	50.10	64.64.2	3658	4917	3380
2.	CrnlRL	10.2	64.64.2	3481	4789	3112

Table 5: Routing evaluation

gives the actual evaluation numbers for the two runs.

For Cornell, DFO does not seem to yield as large of a benefit on the TREC 4 routing task as it did on our TREC 2 and TREC 3 tasks that were reported on for the SIGIR conference. One problem may be that we seriously cut down on the amount of expansion being done, because we weren't sure of the interaction of our new document weighting schemes, the Rocchio algorithm, and DFO. This is probably a mistake since DFO should have ameliorated any poor interaction. Another potential problem is that some of the test set data differed markedly from the learning set data, and DFO may have not handled the new data well.

The CrnlRL run using ITL is an experimental run and not much is to be expected from it. The CrnlRL and CrnlAL official runs are actually the first time the procedures have ever been run after query expansion! Since we had no experience with expansion, we decided to severely limit the expansion for CrnlRL, which undoubtedly led to a lot of the differences between the CrnlRE and CrnlRL runs. We should do some additional runs and evaluations in this area.

It is clear we're not doing a good job taking into account the local term relationships derived from the learning set relevant documents. This information ended up having a very minimal effect on the final ITL similarity; it needs to be featured more prominently.

After the TREC conference, we did some analysis to determine how our results could be improved. Table 6 gives the evaluation of some of those runs. We should have been more trusting of our expansion techniques. Run 2 is the same run as the official CrnlRE run, except doubling the expansion of single terms from 50 to 100, and emphasizing the expanded terms a bit more by decreasing the importance ('A') of the original query weight.

However, the major improvement came as we increased the importance of terms occurring in non-relevant documents('C')! This was a surprise, since with our old weighting schemes like "ltc", the negative information had very little impact. Part of this is due to the length bias associated with cosine weighting. The relevant documents tend to be longer documents whose terms were down-weighted by cosine. With "Ltu" weighted documents in rocchio's formula, the ratio between the contribution of the non-relevant documents to the contribution of the relevant documents was much less (one fourth) as opposed to this ratio when "ltc" weighted documents are used. This is a substantial difference, and suggests that this ratio was being dominated by average length of documents rather than inherent worth of terms.

Once more accurate weights due to relevant and non-relevant document occurrences are obtained, we can decrease the importance of a term being in the original query, and add more expansion terms. Run 3

	Run	*X.Y*	*A.B.C*	R-prec	Total Rel	recall-prec
1.	CrnlRE	50.10	64.64.2	3658	4917	3380
2.	DFO-expmed	100.10	32.64.2	3865	5012	3512
3.	Exphigh	300.30	8.32.256	3724	5348	3489
4.	DFO-exphigh	300.30	8.32.256	4041	5410	3811
5.	CrnlRE-desc	50.10	64.64.2	3777	5111	3547
6.	DFO-high-desc	300.30	8.32.256	4145	5569	3977

Table 6: Post-TREC Analysis

Run	Best	≥ median	< median
CrnlSV(official)	0	22	3
CrnlSE	3	16	6

Table 7: Comparative Spanish Results

	Run	R-prec	Total Rel	recall-prec
1.	CrnlSV(official)	2801	1664	2234
2.	CrnlSE	3118	1748	2821

Table 8: Spanish evaluation

in Table 6 shows expansion by 330 terms and phrases, decreasing the importance of the original query, and increasing the importance of occurrences in the non-relevant documents. This run does not include the DFO optimization, and is already much better than the CrnlRE official run, retrieving an impressive 430 more relevant documents. Adding DFO in Run 4 gives an added improvement; not adding that many more relevant documents, but doing a substantially better job ranking them. This is as expected. The DFO algorithm optimizes performance only among the top documents.

The guidelines of TREC, when read carefully, forbid using certain sections of the text documents, namely those with manually added keywords (e.g. "DESCRIPT") This guideline is not always obeyed, especially by the new groups that are typically overwhelmed with all the other details of their first TREC. Runs 5 and 6 show some of the effects of including those fields. In those two runs, the manually indexed fields were included in the test documents, though not in the learning documents. As can be seen, there is a 5% to 6% improvement over the "legal" runs, which is reasonably substantial. Future TREC's should probably pay more attention to this restriction so that everybody is on an even footing.

Spanish

We did not do much at all for Spanish due to Professor Salton's final illness and then death on August 28th. The Spanish track due date of September 1 precluded any new efforts for Spanish. We attempted to run basically the same two runs as last year, except with our updated weighting approach. The pure vector run worked, but the Spanish expansion run counterpart to CrnlAE had a bug (we invoked the wrong parser, one that was not 8-bit clean). This was tracked down long after the deadline, so only the vector run is an official run, though we present both results.

The CrnlSV run is the pure inner product vector run. The query is weighted with "ltu" weights, and the documents with the "Lnu" weights presented earlier. The CrnlSE run starts off with the CrnlSV results as an initial run. The top 20 documents (without human intervention) are assumed to be relevant. The Rocchio relevance feedback procedure is invoked to expand the original query (50 terms are added), and reweight. This final query is then re-run against the documents, with the top 1000 documents being sent to NIST for evaluation.

SMART is very language independent. The main requirement is that there is some easy method of determining word boundaries, which is true for most, but not all, languages. The total human time for converting SMART to handle Spanish, fashion stemming rules, and forming a stop list was about 5-6 hours (done for TREC 3 [5]).

Spanish Ad-Hoc Results

Table 7 and Table 8 give results for both the official run CrnlSV and the unofficial run CrnlSE. Both did very well compared to the median of all groups doing Spanish, though there was a very significant gap between median and best individual query scores, suggesting one or two other groups did extremely well.

The vector run CrnlSV was above the median for 88% of the queries. The expansion run was above the median for fewer queries, but performed considerably better than CrnlSV for many of those queries. This is as expected; if the initial search is good enough so that the top 20 documents are, if not relevant, at least strongly related to relevant documents, then the reformulated query will work well. However, if the initial documents miss the topic area completely, then the expanded query will go off into the hinterlands, and do very poorly.

One Spanish-specific problem that we had intended to handle this year, but didn't have time for, is that accented characters occur inconsistently throughout the text. There are many cases where the same word occurs both with and without an accent on some letter. There are various transformations that can handle this, and we need to investigate which works well.

Another lack of our Spanish approach that we did not have time to correct, is that phrases are not considered. The SMART approach uses statistical phrases derived from frequently occurring pairs of adjacent non-stopwords. Since there is no linguistic base for the phrases, they can be derived as easily for Spanish as for English. Our English results suggest that an improvement of between 5% and 12% can be achieved when statistical phrases are added (the latter figure resulting from expansion by phrases).

Confusion

We have implemented a two-pass correction algorithm for the confusion track, which seeks to measure the retrieval degradation when the text is highly unreliable, for example with massive OCR errors.

One of the problems associated with traditional OCR correction algorithms is that they strongly depend on having a correct dictionary available, so that mangled words can be mapped onto likely correct words. This can lead to problems when

1. A dictionary is not available.
2. The dictionary coverage does not match the collection.
3. Proper nouns are important.

Our approach does not use a correct dictionary; it only uses the standard collection dictionary of the degraded text. This fits the standard SMART philosophy that the most reliable source of information about a large collection of text is the text itself. This frees SMART from being domain or even language dependent, and greatly reduces the human involvement of working with a new collection of text.

For the first pass, the queries are indexed using the collection dictionary (we assume that each query word occurs correctly at least once in the collection). The query is then expanded by adding all words in the collection dictionary that can be transformed into a query word with one transformation. Here, a transformation can be a deletion, insertion, or substitution of any alphanumeric character. Each one of these added query terms is weighted using an "idf" based upon both the correct query term's collection frequency, and the collection frequency of the transformed term. There are various restrictions on the transformation process, the most important one being a minimum length of the original query term (5 letters).

The TREC document text was degraded using random substitutions of characters, including blanks. The types of errors were therefore random rather than being systematic errors such as would be realistic in an OCR environment. Thus instead of only a few standard mis-scannings of a document word, there tended to be hundreds. Using our process a typical 20 term query is expanded to over 2000 terms, most of which occurred in very few documents!

We earlier experimented with allowing two tranformations per query term, and allowing a query term to be a prefix of a document term (in case the blank between the query term and the next term was deleted), but overall effectiveness was degraded in those preliminary tests. We need to re-examine these possibilities later.

The problems with this first pass approach is that document weights are not being accurately given. The word "antitrust" may occur in 10 different forms within a long document, and a match of each one of those forms is considered an important new piece of evidence. Our first pass retrieval results are thus dominated by long documents.

Run	Best	≥ median	< median
CrnlB	26	20	4
CrnlBc10	24	18	8

Table 9: Comparative Confusion Results

	Run	R-prec	Total Rel	recall-prec
1.	CrnlB	2415	1815	2084
2.	CrnlBc10	1769	1489	1431

Table 10: Confusion evaluation

To correct this, a second correction pass was implemented. The top retrieved documents at the end of the first pass are re-indexed, just as Cornell has been doing for years with our local-global runs. However, instead of using the collection dictionary for the re-indexing, a dictionary consisting of only query terms is used. Each word in a document is compared to this dictionary and is indexed by a query term only if zero or one transformations are required to exactly match the query term. Thus all single-transformation variations of a query term will map to that query term, and the problem of multiple forms of a term is resolved.

The documents are weighted, and a normal inner-product similarity function is applied, resulting in the final similarity value. Note that the document weighting scheme needs to be a slight variation of our normal scheme in that the number of unique tokens in a document can no longer be calculated. Instead, document length is normalized by the total number of tokens in the document.

For our official run, CrnlBc10, we submitted just the results of the first pass query expansion; we ran out of time and couldn't complete the second pass. We hope to have second pass results available by our workshop talk, though they are still not finished. At a later point, the 20% confusion level task will be performed.

Unfortunately, our base run, CrnlB, made on the correct version of the text, is not directly comparable to CrnlBc10. We made it at a time when we were still planning on incorporating stemming into our final confusion run. However, we did not complete (or even start) our stemming work. A fair base comparison would be a run on an unstemmed version of the correct collection, which would probably be 5% to 10% worse than CrnlB.

As Table 9 and Table 10 show, the base case and confusion do very well in comparison with other groups, though the actual level of performance is not that impressive. There are only 3 groups being compared at the 10% confusion level, which meant that an accurate comparison is not really possible. However, for both the base task and the 10% confusion task, our results are very good, giving the best result for roughly half of the queries.

This work is very much a preliminary investigation into the two pass correction approach. We haven't yet implemented a version handling stemming (though plural removal is handled automatically by the transformation algorithm). We expect the standard stemming expansion approach to work, adding any term that stems to the same stem as the query term. Then statistical phrases can be added to at least the second pass, the current approach being single term only. The next step after that is query expansion.

The combination of this two pass approach with the ILT approach outlined earlier offers all kinds of possibilities. A much deeper individual term match can be done (eg, more transformations) if the match can be weighted with the likelihood of the match being correct. If such certainty information is available directly from the OCR system, then it can be used.

Note that this combination correcting-2-pass,ILT approach can be used not only in an OCR environment, but in any retrieval situation where the data is known to be erroneous. The most intriguing possibility is that of speech retrieval, with the initial pass narrowing in on likely "documents" and the second correcting ILT pass using linguistic and other knowledge to decide if a match exists. The important consideration here is that this final analysis is done within the context of the query, which simplifies the task immensely.

Interactive Introduction

In our interactive track participation, we were interested in studying the effectiveness of interactive searching with minimal user intervention. In particular, we wanted to see if simple *relevance feedback* based query modification performs well as compared to a more involved query modification technique where the users can manually add/remove terms to/from the query. In a relevance feedback based interactive search, the users are only asked to judge the presented documents for relevance, a task much simpler than deciding what terms will be appropriate for a search.

We also wanted to test the effect of relevance judgments based on a deep (complete) reading of the presented documents as opposed to a shallow (quick) reading. We submitted two runs, *CrnlI1* – a run based on deep reading of the documents; and *CrnlI2* – a run based on shallow reading of the documents. An interactive search based on shallow reading of documents can be especially useful when the documents marked relevant in the search are used via relevance feedback for query modification and further retrieval of additional documents. It is possible that quick reading of a few documents is sufficient for generating a good search formulation for later use.

Interactive System Design

The searchers were instructed to find as many useful documents for a query as possible in (roughly) twenty minutes. For a query, an interactive session starts with the retrieval of ten documents using the initial query, and proceeds in iterations with obtaining relevance judgments for the ten documents, automatic query modification via relevance feedback, and retrieval of ten new documents using the modified query. Documents retrieved in each iteration are presented to the user in rank order. We implemented a simple, textual user interface with the following features:

- A user can browse a document by either moving forward or backward.
- A user can mark a presented document relevant, non-relevant, or can't decide.
- While viewing a document, a user can also decide to quit the search or get more[1] documents, *i.e.*, go directly to the next iteration. All unseen documents (including the present one) are returned to the potentially retrievable pool.
- A user can also ask to look at the titles from the current iteration. Even though this feature would be useful in systems where searchers are allowed to skip documents by looking at just the document titles, in our system, this feature was almost never used by the searchers. The utility of this feature was also diminished because all TREC documents don't have a well defined title.

One desirable feature that our system misses is the highlighting of the query terms in the documents presented to a searcher. This feature would speed up, and possibly improve the relevance assessment process.

Similar to our routing runs, we used Rocchio's formula to do relevance feedback (with parameters $\alpha = 8$, $\beta = 8$, and $\gamma = 4$). In every iteration, we added *ten* new terms and *two* new phrases to the current query, and reweighted the query according to Rocchio's formula. All documents marked relevant *up to this stage* in the search were used in relevance feedback. This ensures that the relevant documents from the last iteration do not drift the query away from relevance.

Interactive Evaluation

In the following, we refer to the person who conducted the search for us as the *searcher*, and we call the relevance assessor from NIST, the *assessor*.

Interactive Primary Task

[1] more was never used by the searchers in our TREC participation.

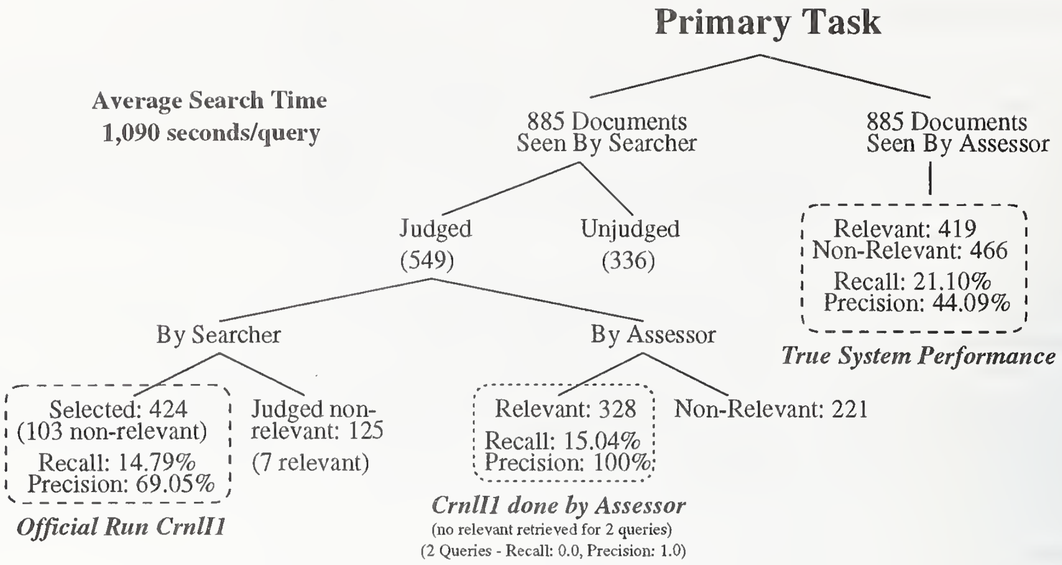


Figure 2: Evaluation of the primary interactive task.

Figure 2 shows the steps involved in official run *CrnlI1* for the primary interactive task. For the 25 queries used in the interactive track, the searcher looked at 885 documents in all. Out of these 885 documents, the searcher could not decide on the relevance for 336 documents, judging a total of 549 (885−336) documents. Out of the 549 documents judged, the searcher thought that 424 were relevant (and selected them for the official submission), and 125 were judged non-relevant. We immediately observe that our searcher's notion of relevance has a noticeable difference from the assessor's notion of relevance. Our searcher marked 103 documents as relevant that are actually non-relevant from the assessor's perspective. In effect, the *precision* of the official run is one measure of the *overlap between the searcher's and the assessor's notions of relevance*, and does not say much about the system performance.

The difference in recall between the assessor doing the search and the searcher doing the search is another measure of the difference in the searcher's and the assessor's view of relevance. We notice that out of the 125 documents that the searcher thought were non-relevant, 7 were actually relevant. Missing these 7 relevant documents marginally lowers the recall for *CrnlI1* (14.79% in place of a possible 15.04%). Of course, if the assessor was the real searcher, the search precision for the selected documents would be 100%. The actual system performance should be measured assuming that the real assessor will be judging *all seen documents* for relevance (there will be no unjudged documents), and that the recall and precision values will be based upon all seen documents as well (not only the ones judged relevant). Assuming similar time requirements in this scenario, the true system performance would have been R: 21.10% and P: 44.09%.

Our searcher marked 103 non-relevant documents as relevant, but missed only 7 relevant documents. In other words, our searcher was usually generous in assigning relevance – many non-relevant documents were selected and not too many relevant documents were missed. As the tendency of searchers to grant relevance can vary considerably from searcher to searcher, the recall and precision precision figures from the evaluation of the primary task are highly subjective to the person doing the search.

Interactive Secondary Task

The secondary task in the interactive track aims at measuring the effectiveness of the final search formulation generated after an interactive search. To generate a final query in the Smart system, we used all documents marked relevant by the searcher in an interactive session, and modified the *original query* via relevance feedback. We expanded the original query by *fifty* terms and *ten* phrases, and used Rocchio's feedback (with parameters $\alpha = 8$, $\beta = 8$, and $\gamma = 4$) for term weight modification. Freezing the documents *judged relevant* by the searcher at the top of the list, removing the documents judged non-relevant by the searcher,

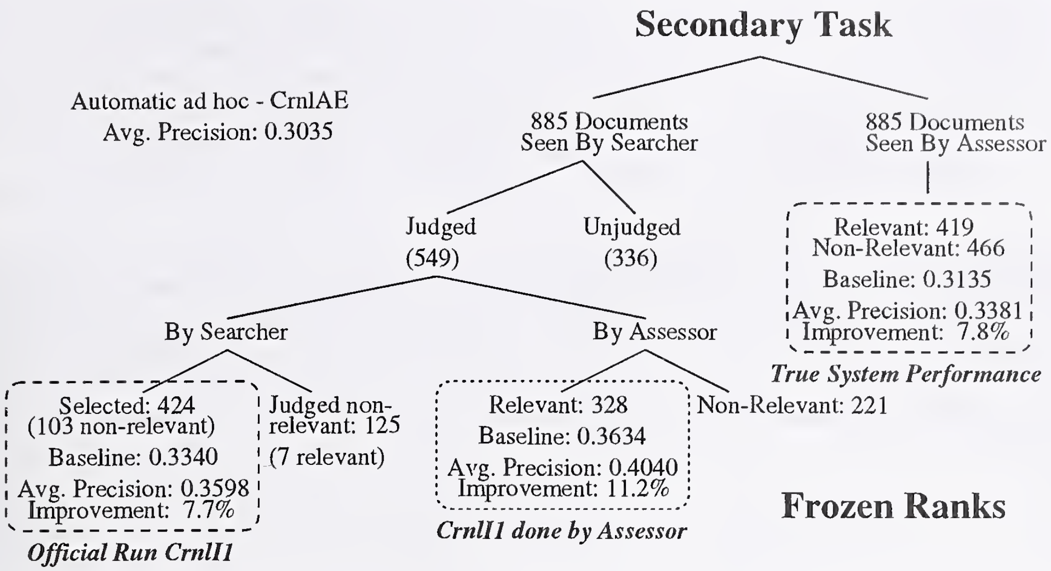


Figure 3: Evaluation of the secondary interactive task.

and returning the unjudged documents to the pool of potentially retrievable documents, we retrieved more documents using the modified query to get a total of 1,000 documents.

When only the documents judged relevant by a searcher are retained in the top ranks, evaluation based on rank freezing becomes highly sensitive to the overlap between the searcher's and the assessor's relevance judgments. Figure 3 shows that if the assessor was doing our official run *CrnlI1*, the average precision would have been 0.4040 in place of 0.3598. The better the overlap between the searcher's and the assessor's judgments, the higher is this figure. But we should observe that by freezing only the *selected documents* at the top, we also improve the baseline of our experiments. For example, if the documents selected by the searcher are frozen at the top, and the base query (from our ad hoc automatic run *CrnlAE*) is used to retrieve more documents until we have 1,000 documents, the average search precision increases from 0.3035 (*CrnlAE*) to 0.3340. If the documents selected by the assessor (actually relevant) are frozen at the top and the base query used to retrieve additional documents, the average precision improves further to 0.3634.

Using rank freezing for only the selected documents, this dependence of average precision on searcher's notion of relevance hinders a direct comparison between the average precision for various participants. A better evaluation of this task would have been the traditional rank freezing evaluation where all seen documents (relevant or non-relevant) are frozen at the top. Using such an evaluation, the true average precision for the run *CrnlI1* is 0.3381.

We evaluated our official run *CrnlI1* by residual collection analysis as well. As the documents retrieved during different iterations depend upon the relevance assessments from the previous iterations, we note that the set of the seen documents can differ if the searcher was running our system, as opposed to if the assessor was running it. To obtain a uniform residual collection for our analysis, we remove any documents that were viewed by a user (searcher/assessor) for a query. Now we can directly compare

1. the base query (from *CrnlAE*),
2. the query generated using the documents marked relevant by the searcher,
3. the query generated if the assessor was marking the documents presented to the searcher (to study the effect of overlap between the searcher's and the assessor's relevance judgments), and
4. the query generated if an assessor was running our system.

	Base (*CrnlAE*)	*CrnlI1*	*CrnlI1* Assessor's Judgments	Assessor Running Smart
Avg. Precision Improvement	0.1302	0.1605 +23.2%	0.1657 +27.3%	0.1681 +29.1%
Exact R-Precision	0.1867	0.2153	0.2185	0.2270

Table 11: Residual collection evaluation of the secondary interactive task.

Run	Avg. Time per Query (seconds)	25 Queries				Average Recall (Micro)	Average Precision (Micro)	Modified Queries Full Coll.
		Total Seen	Total Judged	Judged Relevant	Judged Non-Rel.			
Deep Run *CrnlI1*	1,090	885	549	424 (103 non-rel.)	125 (7 rel.)	0.1479	0.6905	0.3609
Shallow Run *CrnlI2*	854	1,540	970	506 (144 non-rel.)	464 (63 rel.)	0.1378	0.5941	0.3424 –5.1%

Table 12: Deep run vs. shallow run.

Using residual collection analysis, the results in Table 11 show that the difference in the quality of the final search formulation is marginal (.1605 vs. .1681) irrespective of who did the relevance assessments. This fact was not at all apparent from the rank freezing analysis, and suggests the evaluation methodology of future experiments needs to re-considered.

Interactive Deep Run vs. Shallow Run

One of our interactive runs was based on shallow (quick) reading of the documents presented to the searcher. We wanted to test if weak relevance assessments for more documents would result in a better final query as compared to judging fewer documents with in-depth reading of the documents. Results from the deep run (*CrnlI1*) and the shallow run (*CrnlI2*) are compared in Table 12. In the shallow run, the searcher was able to look at many more documents in less time, but the quality of the relevance assessments was not as good as the deep run (average precision fell from 0.6905 to 0.5941).

When more documents are selected (506 in place of 424), one expects the recall to go up. But, surprisingly, we observe a fall in average recall for the shallow run as opposed to the deep run. On further analysis, we find that the average recall is heavily influenced by queries that have very few relevant documents. For example, if we consider only the queries that have at least 25 relevant documents (21 queries), the average recall for the deep run (0.1342) is actually less than the average recall for the shallow run (0.1420). We have observed that such queries, which are generally "hard" for a keyword based retrieval system, oftentimes have relevance buried in few sentences within a relevant document. Such relevance is more amenable to discovery during a deep reading of a document. Therefore, the deep run does much better on such queries.

As the main aim of this experiment was to study the improvement in query quality as a result of a deep/shallow run, we modified the original query, once using all documents judged relevant in the deep run, and then using all documents judged relevant in the shallow run. We searched the *entire collection*[2] using these modified queries. The non-interpolated average precisions from this experiment are shown in the last column of Table 12. We observe that the shallow run has slightly poor performance (5.1% worse) than the deep run, but with the searcher spending less time on a query. Once again, if we remove the queries with fewer than 25 relevant documents, the average precisions for the deep and the shallow run become 0.3675 and 0.3632, respectively. Now the shallow run is almost at par with the deep run (just 1.2% worse).

These results show that if the aim of an interactive session is to improve the quality of the search

[2] The residual collections for the two runs are different.

formulation, a shallow reading of several documents can work almost as well as a deep reading of a few documents, taking less time. We believe that with a better user interface, in particular by highlighting the initial query terms, the quality of the shallow run would benefit more than the deep run, and it is possible that a shallow run might outperform a deep run. We will explore this aspect in our future experiments.

Interactive Note

We discovered a minor mistake in our official interactive track submission. As the search formulation changes over iterations, for some queries our system generated document similarities in later iterations which were greater than the numerical similarity of documents from the previous iterations. As the TREC evaluation programs sort the retrieved documents by similarity, some documents retrieved in the later iterations were placed before some documents retrieved in earlier iterations. This affected the ranking for Q_0 in the secondary task. The primary task was unaffected due to its *set oriented* evaluation. The ranking for documents retrieved in the batch mode was also unaffected. For this reason, the official results for our secondary interactive task are slightly different than the real results.

Run	Official Average Precision	Real Average Precision
CrnlI1	0.3589	0.3598
CrnlI2	0.3225	0.3243

Efficiency

Efficiency issues are becoming increasingly important in these TREC experiments as retrieval methods become more complicated and expensive. Thus it is important to have at least some discussion of efficiency within a paper like this.

SMART is a reasonably fast system. It indexes documents at a rate of about 600–800 megabytes per hour (including inverted files) on a low-end Sun Sparc (Model 20-51). Simple vector retrieval runs can be quite fast. The CrnlVS vector run takes less than a second for all 25 queries combined, when asked to retrieve 10 documents per query. Keeping track of the top 1000 documents is currently much more expensive, adding about 1.3 seconds per query.

The more complicated approaches are much more time consuming, ranging up to 4 minutes per query (CrnlAL). The CrnlAL time for each query includes 3 retrieval passes, re-indexing and relevance feedback expansion of 20 documents, complete re-indexing of 1750 top documents in preparation for the ITL pass, and approximately 50 similarity computations for each of those 1750 documents (one for each query term match within the document).

Luckily, in actual practice the execution times of the complicated methods can be cut down drastically. The massive query expansion approaches will benefit greatly from optimization efforts such as those discussed in our TREC 1 work. Some of the effectiveness increase of the massive query expansion will have to be traded back in order to get reasonable efficiency, but the results of TREC 1 show the effectiveness cost will not be prohibitive. The new approaches like ITL have had no attention paid to optimization, but obviously the efficiency can be improved once they have been proven to be effective.

Comparison with past TREC's

It is difficult to determine how much systems are improving from TREC to TREC since the queries and the documents are changing. For example, in TREC 3 the "Concept" field of the queries was removed. These terms proved to be very good terms for retrieval effectiveness in TREC 1 and TREC 2; thus the TREC 3 task without them is a harder task than previous TRECs. The TREC 4 task was even more difficult since so much more of the text was removed from the queries. This makes the TREC 4 results much more realistic for the ad-hoc retrieval, since most users will type in a sentence at most, but it also depresses the results. To get a handle on how much SMART has improved in the past three years, Table 13 presents the results of running our TREC 1-4 systems on both the TREC 3 and TREC 4 ad-hoc tasks. The automatic approach of

Methodology	Run	TREC 3 Task Rec-Prec	TREC 4 Task Rec-Prec
TREC 1	ntc.ntc	2067 (-)	1538 (-)
TREC 2	lnc.ltc	2842 (+38%)	1627 (+6%)
TREC 3	lnc.ltc-Exp	3419 (+65%)	2012 (+31%)
TREC 4	Lnu.ltu-Exp	3852 (+86%)	2944 (+91%)

Table 13: Comparisons of past approaches with present

SMART has been improving at an average rate of almost 25% per year so far. This rate will probably start tailing off in the future, especially if the queries remain short, but we still expect substantial improvements for next year!

Conclusion

The Cornell SMART Project is again a very active participant in this year's TREC program. With the exception of our interactive track participation, everything we have presented here is completely automatic and uses no outside knowledge base (other than a small list of stopwords to ignore while indexing). Manual aids to the user can be built on top of this system to provide even greater effectiveness.

Our investigations into document length normalization show that the cosine normalization used by SMART for past TRECs is not well suited for full text documents. The "Lnu" weighting scheme presented here handles the TREC documents with much less of a length bias. This is very important for the short TREC 4 type of queries, which are strongly affected by normalization issues.

Our ad hoc expansion approach, exemplified by the CrnlAE run, works very well. Queries are expanded by terms occurring in the top retrieved documents, and reweighted using the Rocchio relevance feedback formula.

Our Individual Term Locality (ITL) similarity approach attempts to come up with a new similarity function operating on individual term occurrences instead of the normal vector representation of the document. This holds great potential for development in the future, since non-statistical information about individual terms can easily be used within the model.

Our routing run this year performed unspectacularly (right about the median); we haven't been able to analyze why. The Dynamic Feedback Optimization approach used very successfully in our experiments for SIGIR did not perform well here.

Our interactive results showed that minimal involvement by users, just having users judge the relevance of documents, can result in a very effective retrieval set.

We tried a new 2-pass dictionaryless correction algorithm for the confusion (OCR) track. All "correction transformations" were made between terms occuring in the erroneous documents, and the query terms, no correct dictionary was involved. This performed very well, at least as well as the other entries in the track (CrnlBc10 had the best results on half the queries), even though only the first half of the algorithm was actually run.

The Spanish results were again considerably above the median, and used no language knowledge. The runs were exactly the same runs as we run on the English tasks.

A comparison with previous years' TREC approaches show that SMART is averaging about a 25% improvement per year. That rate will be difficult to maintain, but we'll try!

References

[1] Chris Buckley. *Massive Query Expansion for Relevance Feedback.* Cornell University, 1995.

[2] Chris Buckley, James Allan, and Gerard Salton. Automatic routing and ad-hoc retrieval using SMART : TREC 2. In D. K. Harman, editor, *Proceedings of the Second Text REtrieval Conference (TREC-2)*, pages 45–56. NIST Special Publication 500-215, March 1994.

[3] Chris Buckley and Gerard Salton. Optimization of relevance feedback weights. In Ed Fox, Peter Ingwersen, and Raya Fidel, editors, *Proceedings of the Eighteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 351–357, New York, July 1995. ACM.

[4] Chris Buckley, Gerard Salton, and James Allan. Automatic retrieval with locality information using SMART. In D. K. Harman, editor, *Proceedings of the First Text REtrieval Conference (TREC-1)*, pages 59–72. NIST Special Publication 500-207, March 1993.

[5] Chris Buckley, Gerard Salton, James Allan, and Amit Singhal. Automatic query expansion using SMART : TREC 3. In D. K. Harman, editor, *Proceedings of the Third Text REtrieval Conference (TREC-3)*. NIST Special Publication 500-225, 1995.

[6] J.J. Rocchio. Relevance feedback in information retrieval. In Gerard Salton, editor, *The SMART Retrieval System—Experiments in Automatic Document Processing*, chapter 14. Prentice Hall, Englewood Cliffs, NJ, 1971.

[7] Gerard Salton. *Automatic Text Processing — the Transformation, Analysis and Retrieval of Information by Computer.* Addison-Wesley Publishing Co., Reading, MA, 1989.

[8] Gerard Salton. Developments in automatic text retrieval. *Science*, 253:974–980, August 1991.

[9] Gerard Salton and Chris Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523, 1988.

[10] Gerard Salton and Chris Buckley. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 41(4):288–297, 1990.

[11] Amit Singhal, Gerard Salton, Mandar Mitra, and Chris Buckley. Document length normalization. Technical Report TR95-1529, Cornell University, 1995.

Appendix:Interactive system description

0.1 Experimental Conditions

1. **Searcher Characteristics**

   (a) Number of searchers in experiment: 2

   (b) Number of searchers per topic: 2

   (c) Age of searchers: 25 and 27

   (d) IR searching experience of searchers: None

   (e) Educational level of searchers: Ph.D. Students

   (f) Undergraduate major of searchers: Computer Science

   (g) Experience/familiarity with subject of topic: Variable (per topic)

   (h) Work affiliation of searchers: Department of Computer Science, Cornell University

2. **Task Description:** Find as many documents as possible that (you think) are relevant to a topic in approximately *twenty minutes.*

3. **Training**

   (a) Description of the training process: In the training phase, the searchers used the search interface to conduct searches for old TREC topics (selected topics from TREC topics 1–200).

   (b) Time for training: Searcher 1 - (approx.) 240 minutes, Searcher 2 - (approx.) 195 minutes.

0.2 Search Process

1. **Clock Time**

	Seconds per Topic			
Run	Mean	Median	Std. Dev.	Range
CrnlI1	1,090	1,051	220	649 – 1,469
CrnlI2	854	859	180	471 – 1,165

2. **Documents Viewed**

   (a) **Viewing:** A document is considered viewed if any part of it, *other than the title,* is presented to a searcher.

   (b) Number of documents viewed:

	Documents per Topic			
Run	Mean	Median	Std. Dev.	Range
CrnlI1	35.40	33	9.76	19 – 53
CrnlI2	61.60	66	15.22	30 – 90

3. **Iterations**

   (a) **Iteration:** Retrieve and display ten documents using the query from the previous iteration (initial query used in the first iteration), obtain relevance judgments for these documents, and modify query using these relevance judgments.

   The searcher can end any iteration by asking to do another iteration or by quiting the search, without looking at all ten documents. In this case, the unseen documents are returned to the potentially retrievable pool. Even if the searcher looks at just one document presented in an iteration, we consider it a full iteration.

   (b) Number of iterations:

Run	Iterations per Topic			
	Mean	Median	Std. Dev.	Range
CrnlI1	4	4	1.08	2 – 6
CrnlI2	6.32	7	1.63	3 – 9

4. **Number of Search Terms**

We consider a single term or a phrase as a *term*. For example, topic 203 has five terms initially:

- Single terms: **tire**, **econom**, **impact**, **recycl**
- Phrase: **econom impact**

(a) Number of terms in initial query:

Run	Initial Terms per Topic			
	Mean	Median	Std. Dev.	Range
CrnlI1	9.36	8	4.18	3 – 17
CrnlI2	Same as *CrnlI1*			

(b) Number of terms in final query:

To obtain a final query we added fifty new single terms and ten new phrases to the *initial query* via relevance feedback, based upon the the searcher's relevance assessments.

Run	Final Terms per Topic			
	Mean	Median	Std. Dev.	Range
CrnlI1	69.36	68	4.18	63 – 77
CrnlI2	Same as *CrnlI1*			

5. **System Features**

The following system features were used by the searchers whenever needed:

f	Forward: Browse document forward one page (35 lines)
b	Back: Browse document backward one page (35 lines)
y/r	Rel: Mark document relevant
n	Non-Rel: Mark document non-relevant
c	No Clue: Mark document can't decide
t	Titles: Show titles from current iteration (seldom used)
m	More: Get more documents by going to next iteration directly (never used)
q	Quit: Quit search

6. **Errors**

Not applicable.

7. **Narrative for Topic 236**

(a) **Set-Up**

The search was conducted using 2 windows — one displayed the query text, and was adjusted such that only a single query could be displayed in it at a time; the actual search was conducted in the other window.

The windows looked typically like the following:

- **Search Window:**

```
Smart (ntq?): Trec 236
Num  Action Sim Title
875162      19.51 COMMANDERS SAY WOMEN SHOULDN'T BE IN COMBAT FEMALE OFFICERS PRE
543651      19.21 With PM-Welfare Overhaul Bjt</HEAD> WORK</HEAD> CHILD CARE</HEA
875966      18.61 NEW FORMULA JACKS UP COST OF RENEWING LICENSE TAGS </HEADLINE>
560047      18.02
754212      17.87 Business, Administration Support National Product Liability Law
845980      17.79 GOVERNOR SIGNS BILL LIMITING TOBACCO-SAMPLE GIVEAWAYS </HEADLIN
778338      17.75 Congressman Wants More Regulations For Cosmetics</HEAD>
576988      17.29 U.S. Votes Against Law Of Sea Convention</HEAD>
558522      16.99 Wetlands Threatened by Water-Level Increases</HEAD>
857354      16.79 A WILD RIDE OFF THE CANADA COAST </HEADLINE>
Hit return to continue ...
```

- **Query Window:**

```
<num> Number:  236

<desc> Description:

Are current laws of the sea uniform?  If not, what
are some of the areas of disagreement?

</top>

<top>

<num> Number:  237
```

(b) **Search Process**

The search proceeds thus:

i. At the system prompt we enter "**Trec 236**". The system logs the current time in a log file as:

```
Trace: entering trec_pager
Elapsed Time:   34387.345579
```

and retrieves 10 documents in response to query 236 and displays them in the search window as shown above. While the system is retrieving documents, we read the query-text. Since the queries are very short, this takes very little time.

ii. On typing return, the text of the top-ranked document is displayed by the pager. The screen now looks like this [3]:

```
857354      16.79 A WILD RIDE OFF THE CANADA COAST </HEADLINE>
Hit return to continue ...
.- --- 875162 /fsys/thor/k/trec.d3/sjm/sjm_158 977136 984193

.n 13 39
 SJMN91-06171051 </DOCNO>

.w 76 207
  Chart, photo; PHOTO: Associated Press; Maj. Christine Prewitt, right, and Lt.
```

[3] We show successive screens with an overlap of 2 lines so that the reader may easily follow the sequence of actions.

```
Brenda Marie Holdener testify (color).  </CAPTION>

.s 305 734
  The Pentagon's senior officers insisted Tuesday that current rules barring
women from serving in combat should continue, while some members of the Senate
Armed Services Committee heartily endorsed allowing women to fly warplanes in
combat.;    The crowded hearing marked the first formal congressional inquiry
into allowing women to fly combat aircraft since the Persian Gulf war, where
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@ f:Forward, b:Back, y/r:Rel, n:Non-Rel, c:No Clue, t:Titles, m:More, q:Quit
```

f and **b** are used to move a page forward or backward in the document text. When we have read enough of the document to be able to judge its relevance, we type the appropriate letter (**y**/**n**/**c**). Note that, we avoid reading the entire document only when relevance is clearly established before the end of the document. **c** is used when we are unable to decide the document's relevance.

iii. When a relevance judgement is entered, the pager automatically displays the next document. Thus, when the first document has been viewed (we were unable to decide whether this particular document was relevant or not), and the appropriate judgement (**c**) entered, the screen looks like this:

```
.w 6498 6526
<CITY>  Washington  </CITY>
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@ f:Forward, b:Back, y/r:Rel, n:Non-Rel, c:No Clue, t:Titles, m:More, q:Quit c
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
.- --- 543651 /fsys/thor/k/trec.d2/ap/ap880617 154899 158945

.n 13 37
 AP880617-0048 </DOCNO>

.t 191 227
With PM-Welfare Overhaul Bjt</HEAD>

.w 271 559

   Here is a comparison of the current welfare
system and the welfare overhaul bills passed by the House and
Senate. Negotiators will reconcile differences in the two bills
before sending a measure to the White House, where it is not clear
whether President Reagan will sign it.
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@ f:Forward, b:Back, y/r:Rel, n:Non-Rel, c:No Clue, t:Titles, m:More, q:Quit
```

iv. This continues until the last document retrieved in an iteration has been judged. At this stage, the system internally modifies the query using relevance feedback and returns 10 more documents and displays them as before. The whole process (steps (ii) to (iv)) is repeated for the new iteration.

v. Finally, when we exhaust the allotted time for a query or feel that we have examined a sufficient number of documents, we press **q**, and return to the system prompt. The system logs the current time again to mark the end of the search:

```
Trace: leaving trec_pager
Elapsed Time:   35615.352744
```

The documents retrieved after the first iteration were:

```
Num  Action Sim Title
807542     2.24 Conference Considers Ban on Toxic Ocean Dumping; US Wants More
808752     2.20 Global Accord Reached to Clean Up World's Oceans</HEAD>
590029     2.16 U.S. Territorial Waters Extended From Three to 12 Miles</HEAD>
766572     2.10 U.S. Safety Investigators Want More Access To Cruise Ships</HEA
598342     2.08
517530     2.03 Twenty-Three Nations Sign Treaty to Combat Terrorism at Sea</HE
575157     1.98 U.S.-Soviet Negotiators Call For Halt To Overfishing</HEAD>
779976     1.97 US-Soviet Accord on Bering Sea Producing No Bonanzas</HEAD>
605081     1.95
515744     1.90 U.S. Law To Close PLO Mission Assailed By General Assembly</HEA
Hit return to continue ...
```

The documents retrieved after the second iteration were:

```
Num  Action Sim Title
795682     4.10 Child Rights Convention Comes Into Force</HEAD>
753624     3.94 UN Panel Approves Measure Opposed by U.S., Panama, Others</HEAD
529790     3.92 World Court Rules Against Washington Over PLO Office</HEAD> <NO
538572     3.90 With PM-Summit-Reagan Bjt</HEAD> Reagan, Gorbachev Expected To
588328     3.84 U.N. Conference Adopts Convention Against Illicit Drug Traffick
521133     3.84 U.N. General Assembly Slaps United States For PLO Eviction Atte
811205     3.84 U.S. Negotiators to Push Limited Mining Moratorium</HEAD>
575767     3.84 U.S. Citizens Group Says U.S.-U.N. Relations in Disarray</HEAD>
590177     3.81 U.S. Official Says New 12-Mile Offshore Limit Will Help Deter S
794140     3.81 Experts Say Iraq's Actions Against Embassies Violate Internatio
Hit return to continue ...
```

Thus, for this query, we ran 2 iterations after the initial retrieval, and examined 29 of the 30 documents retrieved. 10 documents were judged non-relevant, 6 were deemed relevant, and we were unable to decide 13 documents. All this took 1228.007165 seconds (20.47 mins.).

Recent Experiments with INQUERY

James Allan, Lisa Ballesteros, James P. Callan, W. Bruce Croft, and Zhihong Lu
Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts
Amherst, Massachusetts 01003, USA

Past TREC experiments by the University of Massachusetts have focused primarily on ad-hoc query creation. Substantial effort was directed towards automatically translating TREC topics into queries, using a set of simple heuristics and query expansion. Less emphasis was placed on the routing task, although results were generally good. The Spanish experiments in TREC-3 concentrated on simple indexing, sophisticated stemming, and simple methods of creating queries.

The TREC-4 experiments were a departure from the past. The ad-hoc experiments involved "fine tuning" existing approaches, and modifications to the INQUERY term weighting algorithm. However, much of the research focus in TREC-4 was on the routing, Spanish, and collection merging experiments. These tracks more closely match our broader research interests in document routing, document filtering, distributed IR, and multilingual retrieval.

The University of Massachusetts' experiments were conducted with version 3.0 of the INQUERY information retrieval system. INQUERY is based on the Bayesian inference network retrieval model. It is described elsewhere [7, 5, 12, 11], so this paper focuses on relevant differences to the previously published algorithms.

1 Description of Ad-Hoc Experiments

For the ad-hoc retrieval experiments, the major change to the system was a new estimation technique for term weighting. We also continued to refine our analysis techniques for the TREC topics, our use of passage retrieval, and query expansion using InFinder[1].

1.1 Query Processing

TREC topics 201–250 differ from earlier TREC topics in that the <title> fields were removed. This change makes the TREC topics even more dissimilar from user queries in an online system than in the past. The TREC topics observe the niceties of grammar, punctuation and, especially, polite periphrasis. In an online system, users typically discard grammar, punctuation and any non-functional verbiage in an effort to get the information they want.

The removal of the <title> field created a set of topics that resembled essay questions. Much of our TREC processing this year focussed on creating queries that more closely resemble "real" online queries, by stripping off the polite circumlocution and its accompanying grammar. As a result, in addition to the standard "stop-phrase" program distributed with INQUERY, which removes the occasional polite circumlocution, we resurrected an old program for removing additional verbiage that is likely to be content-free, especially in questions. For example, in topic 201

[1] Formerly called PhraseFinder.

> "What procedures should be implemented to insure that proper care is given to children placed under the au pairs' responsibility?"

only the phrase "au pair" is actually useful. Our second stage stop-phrase processing removes "what procedures should be implemented," which gives a small improvement in performance, but it is unable to strip the topic down to just the single phrase.

Besides intensified stop-phrase processing, the only innovations this year were slightly improved part-of-speech tagging (we used JTAG [14]), and removal of references to the U.S. (in the past we have alternately removed them or downweighted them).

The complete topic-to-query process consisted of the following processes in the order specified.

1. Produce words:
   (a) Remove stop-phrases.
   (b) Remove additional stop-phrases.
   (c) Remove references to U.S., generalize references to U.K, U.N., U.S.S.R and turn two-word country names into #PHRASE.
2. Produce phrases:
   (a) Tag for part of speech.
   (b) Remove stop-phrases.
   (c) Remove additional stop-phrases
   (d) Make noun groups into #PHRASES: N1 N2 N3 ⟶ #PHRASE(N1 N2) #PHRASE (N2 N3).
   (e) Remove references to U.S., generalize references to U.K, U.N., U.S.S.R and turn two-word country names into #PHRASE.
   (f) Remove company suffixes, such as "Inc."
   (g) Discard anything that does not form a #PHRASE.
3. Combine words and phrases for each query, using a #SUM operator.

1.2 InFinder

InFinder[2] is a technique for corpus-based query expansion [8, 1, 4]. For TREC-4, a subset of 30% of the adhoc document set was used to build the InFinder database. None of the Federal Register and Patent documents were used. Noun phrases were defined as being all single, doubles, and triples of adjacent nouns as determined by the JTAG part of speech tagger [14]. Concepts that occurred less than 16 times or greater than 3,000 times were eliminated. Terms that co-occured with a noun phrase more than 20,000 times were also eliminated.

Queries, created as described above, were used to retrieve phrases from the InFinder database. The 30 most highly ranked noun phrases were added to the queries. When adding noun phrases, each phrase was enclosed in a #3 operator and given a weight that reflected whether all terms in the noun phrase occured in the original query (called a "duplicate" phrase) or contained some or all new terms (called a "novel" phrase).

[2] Formerly called PhraseFinder.

Duplicate phrases were given a weight of $1 - \frac{1}{30} \cdot D$, where D was the number of duplicate phrases that preceded it in the phrase ranking. Thus, the first duplicate phrase was given a weight of 1.0, the second was 0.967, the third 0.933 and so forth.

Similarly, novel phrases were given a weight of $0.3 - \frac{0.3}{30} \cdot N$, where N was the number of novel phrases that preceded it in the phrase ranking. Thus, the first novel phrase was given a weight of 0.3, the second was 0.29, the third was 0.28 and so forth.

The weights used were a function of rank with respect to phrases of the same type (novel or duplicate) only. For instance, the first novel phrase would have a weight of 0.3 even if it followed 29 duplicate phrases in the overall ranking. Only the top 30 phrases were added, regardless of type.

1.3 Passage Retrieval

In [3, 4], we reported experiments that showed significant improvements in retrieval effectiveness when document rankings based on the entire document text are combined with rankings based on the best passages in the documents. The TREC-4 ad-hoc document retrieval experiments tested a new approach to using passages in retrieval. The queries used for retrieval were of the form

#SUM (#SUM (Q') #PASSAGE200 (Q))

where Q is a query created from the TREC topic (Section 1.1) and Q' is the expanded form of Q (Section 1.2).

In the past, passage-level evidence was weighted more heavily than document-level evidence. However, the new term-weighting formula (described below) improved the quality of document-level evidence sufficiently that even weighting seemed more appropriate.

1.4 Manual Modifications

These queries were generated by simulating some of the modifications a user might make to an initial query in an interactive environment. The starting point for this experiment was the automatically produced queries (INQ201, described above). Changes to these initial queries were limited to the deletion of spurious (in the user's opinion) words and phrases, modification of weights based on perceived relative importance, and adding proximity restrictions such as are often used in Boolean systems. The user spent an average of 2-3 minutes per query (2 hours for 50 queries).

1.5 Estimation

INQUERY relies on a *tf.idf* formula for estimating the probability that a document is about a concept. The estimation formulae have been used for several years, with only minor modifications [1, 5, 12, 11]. In TREC-3, we began experimenting with new formulae for the adhoc runs. The TREC-3 formulae offered only minor improvements over the more traditional formulae [1], so they were discarded.

Experiments prior to TREC-4 suggested that the Okapi treatment of tf [9] was most effective with common terms, while the INQUERY treatment of tf was most effective with infrequent terms. For TREC-4, we adopted a term weighting formula that is a combination of the two.

$$\begin{aligned} idf &= \frac{\log(\frac{C+0.5}{df})}{\log(C+1.0)} \\ T_i &= \frac{\log(tf+0.5)}{\log(max\_tf+1.0)} \end{aligned}$$

$$
\begin{aligned}
T_o &= \frac{tf}{tf + 0.5 + 1.5 \cdot \frac{L}{avg\_doclen}} \\
ntf &= d_t + (1 - d_t) \cdot (idf \cdot T_i + (1.0 - idf) \cdot T_o) \\
d_t &= \frac{\log(avg\_doclen)}{24} \cdot e^{-5 \cdot \frac{df}{C}} \\
\text{bel}_{\text{term}}(Q) &= 0.4 + 0.6 \cdot ntf \cdot idf
\end{aligned}
$$

where

L = the length of the document (in words, including stopwords),
tf = the frequency of term t in the document,
max_tf = the frequency of the most frequent term in the document,
df = the number of documents in which term t occurs, and
C = the number of documents in the collection.

T_i is the usual INQUERY tf weight, while T_o is the Okapi tf weight.

d_t is the minimum term frequency component when a term occurs in a document. d_t was set to 0.4 in the past. However, the appropriate value appeared to be collection-dependent, and rarely 0.4. A goal of our research was to identify automatic methods of determining d_t for a given collection. The method used in TREC-4 was based on document length, and the frequency of the term in the collection. Either of these factors makes some sense on its own. However, the combination was discovered by accident, and is difficult to justify.

The "good" part of this combination is that d_t depends upon the average document length in the collection. In other words, the average document length controls the importance of a ±1 variation in tf.

The "bad" part of this combination is that d_t depends partly upon the frequency of the term in the collection. It is unclear why such an "idf-style" statistic should be part of the d_t.

These adjustments to the estimation formula were tested against more than just the TREC document collection. In experiments prior to TREC-4, they were found to yield improvements at all levels of recall on the CACM (2 query sets), West FSupp (2 query sets), NPL, and TREC (2 query sets) document collections. These collections vary widely in number of documents and average document length, suggesting that the new formula might be relatively robust.

2 Description of the Routing Experiments

The routing experiments for TREC-4 were an extension of past routing efforts and an incorporation of new ideas inspired by other TREC research groups. Queries were expanded by adding terms, adjacent word pairs, and nearby word pairs. The selected concepts were chosen from a large candidate set by comparing their occurrences in relevant and non-relevant training documents. Weights were assigned using the Rocchio formula applied to INQUERY's version 2.1 weighting scheme. Finally, the weights were adjusted by fitting them more closely to the training data using a technique very similar to one described by Buckley and Salton [2].

2.1 Term selection

The training data for the routing queries consisted of all known relevant documents in TREC disks 1–3, and the same number of top-ranked non-relevant documents retrieved by the original query on that database. (Non-relevant documents are those not judged relevant; they may not have been specifically judged non-relevant. The "original query" refers to the result of creating an INQUERY structured query from the original TREC topic.)

For each query, all terms occurring in the relevant documents were identified, and were then ranked by their relative occurrences in the relevant and non-relevant documents. That is, by:

$$\frac{df_{rel}}{n_r} - \frac{df_{nonrel}}{n_{nr}}$$

where df_{rel} is the total number of relevant documents containing the term, df_{nonrel} is that count in non-relevant documents, n_r is the number of relevant documents, and n_{nr} is the number of non-relevant. The top 50 *non-query* terms in that order were chosen and weighted using a Rocchio formula:

$$\beta \cdot \frac{1}{n_r} \sum_{rel} belief - \gamma \cdot \frac{1}{n_{nr}} \sum_{nonrel} belief$$

where $\beta = 2$, $\gamma = \frac{1}{2}$, and the belief for term t in doc d was calculated by the formula:

$$0.4 + 0.6 \cdot (0.4 \cdot \min(1, \frac{200}{maxtf_d}) + 0.6 \frac{\log(tf_{t,d} + 0.5)}{\log(maxtf_d + 1)}) \cdot \frac{\log((n_t + 0.5)/N)}{\log(N+1)}$$

where $tf_{t,d}$ is the number of occurrences of term t in document d, $maxtf_d$ is the largest number of times any term occurs in documents d, n_t is the number of documents in the collection containing term t, and N is the total number of documents in the collection. This equation is the belief function used by INQUERY version 2.1.

2.2 Additional concepts

The same process described above was applied to find concepts based upon pairs of terms also. In this case, candidate pairs were found considering *only* the 200-word passage of the training document which best matched the original query. From those passages, 50 adjacent term pairs (ordering significant) were chosen. In addition, 50 each of word pairs within 5, 20, or 50 words (order *in*significant) were added. Selection and weighting were done exactly as described above.

In all, each query was augmented with 250 new concepts, though there was some overlap. In query 3, for example, "joint venture" appeared in every category.

The original query and additional concepts were combined in two ways. For official run INQ204, they were put together into a new query of the form:

```
#wsum( 1.0   1.0 original-query
             1.0 50-terms
             1.0 50-#1
             1.0 50-#uw5
             1.0 50-#uw20
             1.0 50-#uw50 )
```

Note that the original query, and each set of 50 new concepts all received the same significance in the query. (The next section mentions how that is changed.) The Rocchio weights for the concepts were incorporated within each group, so the weights balanced term against term, but not term against pair.

For run INQ203 all 250 additional concepts were added at the same level, meaning that their Rocchio weights were deciding the significance of terms and pairs relative to each other. In addition, the original query structure was flattened so that its components were balanced directly against the new concepts.

2.3 Weight adjustments

Inspired by the Dynamic Feedback Optimization approach of Buckley and Salton[2] (which was in turn inspired by the term selection method of City University[9]), we adjusted the chosen weights to achieve higher effectiveness in the training data, predicting that this effort will result in better effectiveness in the test documents.

The approach starts by evaluating the query on the training data. Then some concept weight is adjusted and the slightly different query is evaluated. If the effectiveness has improved, the change is retained; if the new weight hurts effectiveness, the original is restored. In both cases, the next concept weight is tried. This process repeats until no improvement is made.

The reweighting algorithm operated at the top-level of the query network only. For INQ203 that meant that each concept or query element could have its weight adjusted; for INQ204, the balance between the query and the various sets of concepts could change.

For efficiency reasons, the evaluations were done using only the 5000 documents retrieved in response to the new query (prior to reweighting). Weights were adjusted in 5 passes, with factors of 2.0, 1.5, 1.25, 1.125, and 1.0625. In each pass, each concept or query element was potentially reweighted by $w_{new} = w_{prev} \cdot \mathit{pass\_factor}$. Unlike Buckley and Salton's technique, a pass was continued until no concept's reweighting improved results. The purpose of stopping sooner than that is to avoid overfitting the training data; however, the better fitting was preferable during tests on other databases.

3 Description of the Spanish Experiments

The Spanish retrieval experiments built upon the Spanish work done in TREC-3 [1]. This year's effort incorporated InFinder [8] for query expansion, and focused on comparing INQUERY 2.0 (used in TREC-3) with the modified version of INQUERY 3.0 used for the English adhoc experiments.

3.1 Query Processing

Query processing for the Spanish topics is similar to that of the English topics and was used to generate base queries for retrieval. We do not have a Spanish part-of-speech tagger, but the text in the Spanish topics was analyzed with a simple noun phrase recognizer. Sequences of nouns and noun-adjective pairs were chosen for the #PHRASE operator.

The English stop phrase heuristics have not been translated into Spanish, but a few simple stop phrases were removed automatically. A list of the discarded stop phrases is given in Table 1.

Spanish	English translation
evidencia de	evidence of
hay	are/is there
indicaciones de	indications of
cuáles son	which are
cómo van	how is
tendrá	will it be/have
información sobre	information about

Table 1: Stop phrases removed from Spanish topics.

3.2 Noun Phrase Recognizer

Our noun phrase recognizer uses morphological rules to identify words that are likely to be nouns. Sequences of capitalized words are suggestive of proper nouns and some word endings are indicative that a word is likely to be a noun. For example, nearly all Spanish words ending in "d" are nouns [10]. Eleven types of endings were used to identify possible nouns and are given in Table 2.

Ending	Example
-dor	matador (bull fighter)
-d	verdad (truth)
-ata	corbata (tie)
-z	arroz (rice)
-[sz]mo	capitalismo (capitalism)
-miento	conocimiento (knowledge)
-[cs][ií]a	democracia (democracy)
-[cgnstx]i[oó]n	lección (lesson)
-az[oó]n	corazon (heart)
-cida	conocida (acquaintance)
-i[ae]nte	pariente (relative)

Table 2: Spanish Noun Word Endings

There is some ambiguity in the use of word endings as a heuristic to identify nouns. The word "denuncia" can be the noun "report" or can mean "he/she/you are reporting". Therefore we employ syntactic rules to reduce the ambiguity. For example, we require that a definite article precede a noun. In the case of the -cia ending, it is possible for the rule to fail to correctly classify a word. *La denuncia* can mean *the report*, but *No la denuncia* translates to *he/she/you is/are not reporting her*. The syntactic rules are heuristics and may not determine the part of speech to which a word belongs, but they increase the probability that a word is a noun.

There are several ways to modify a noun in Spanish. Qualitative adjectives generally follow nouns while quantitative adjectives precede them. Prepositions may also be used in a noun adjective phrase. For example, *trabajos de repavimentación* can be said to mean "repavement work" where "de" is a preposition meaning "of". The recognizer contains rules to recognize these phrasal constructs, in addition to recognizing nouns and proper nouns.

3.3 Query Formation

The nouns and noun phrases selected by the recognizer were used in lieu of tagged text, to identify phrasal concepts for InFinder. For the Spanish retrieval experiments, an InFinder database was created from the entire 208 MB INFOSEL collection. Table 3 shows a sample query and the top 20 phrases returned for it. Spanish queries were expanded, using the Spanish InFinder database, with the same techniques described in Section 1.2 for expanding English queries.

The final query form combined document-level and passage-level evidence, as was done for the English experiments (Section 1.3). The Spanish experiments were conducted with two different term weighting algorithms, apparently changing the relative worth of document-level and passage-level evidence. Query set SIN010, which was run against INQUERY 2.1, gave passage-level evidence twice the weight of document-level evidence. Query set SIN011, which was run against INQUERY 3.0 using the modified term weighting formulae described above, weighted them evenly.

Query:	indicaciones de las relaciones económicas y comerciales de México con los paises europeos	indications of the economic and comercial relations between Mexico and the European countries

belief	Phrase	Translation
0.486925	relaciones comerciales	commercial relations
0.485157	relaciones economicas	economic relations
0.483628	naciones europeas	european nations
0.479479	cuenca del pacífico	pacific basin
0.478936	comunidad economica europea	european economic community
0.478386	comunidad europea	european community
0.478051	jacques delors	president of the EEC
0.476194	ronda del uruguay	round of talks (GATT) in Uruguay
0.475229	comunidad europea	european community
0.474429	países europeos	european countries
0.474275	asuntos mundiales	world affairs
0.474199	barreras comerciales	commercial barrier
0.474028	grupo de río	river group
0.473967	barros valero	subsecretary of exterior relations
0.473196	lazos comerciales	commercial ties
0.472589	economico comerciales	incomplete phrase, probably was “relaciones económico comerciales”
0.472376	cancilleres	chancellor
0.472358	gary hufbauer	investigator for the Institute for International Economies
0.472293	acuerdos comerciales	comercial agreements
0.471928	viceministros	vice-ministers

Table 3: Sample Query and Top 20 InFinder Phrases

4 Description of the Collection-Merging Experiments

In the collection merging track, the document collection was divided by source and/or date into 10 smaller document collections. The goal of the collection merging track was to select one or more collections to search for a given query, to search, and then to merge the document rankings returned into a single consistent set of rankings.

Our experiments were all based on testing variations of the techniques described in [6]. Five experiments were conducted, labeled INQ206 – INQ209. Each experiment was conducted with the INQ201 query set created for the adhoc track. The collection ranking and results merging algorithms were varied, as described below.

INQ207: A previously published method [6]. The “traditional” INQUERY term weighting formulae were used.

INQ208: Similar to INQ207, except that prior to merging document rankings, a document's score was normalized based on the minimum and maximum possible scores a document could obtain in that collection for that query.

INQ205: Similar to INQ207, except that the “modified” term weighting formulae (Section 1.5) were used, which required minor modifications to the collection ranking algorithm.

INQ206: Similar to INQ208, except that the "modified" term weighting formulae (Section 1.5) were used, which required minor modifications to the collection ranking algorithm.

INQ209: Similar to INQ206, except that the merging algorithm used only the collection's score and the document rank with respect to its collection, i.e. the document's score was *not* used.

Briefly, the goal for INQ208 was to produce a more normalized document score from each collection. INQ205 and INQ206 replicated INQ207 and INQ208, but with the "modified" term weighting function discussed in Section 1.5. INQ209 investigated what could be accomplished if less information were available for merging rankings.

5 Ad-Hoc Results and Discussion

Two sets of results, INQ201 and INQ202, were evaluated in the ad-hoc document retrieval evaluation. The INQ201 results were based on completely automatic processing of the TREC topic statement into a query, automatic query expansion, use of passage-level and document-level evidence, and adjustments to INQUERY's estimation formula. INQ202 was a semi-automatic experiment in which a user was allowed to edit the INQ201 query prior to submitting it to NIST.

The official results for INQ201 and INQ202 are summarized below.

Query Type	**Average Precision**			
	5 Docs	**30 Docs**	**100 Docs**	**11-Pt Avg**
INQ201	.51	.35	.25	.24
INQ202	.60 (+16.8%)	.44 (+25.4%)	.31 (+22.1%)	.29 (+21.0%)

Limited user modification of INQ201 produced a very significant 21.0% improvement in average precision. Most of this improvement appears to be due to 1) deleting useless query terms introduced by query processing or InFinder, and 2) grouping query terms with proximity operators. Clearly there remains room for improvement in automatic query processing techniques.

Query expansion with InFinder was much less effective than in the past. The inclusion of InFinder terms in the query yielded a 3.5% improvement in average precision (Table 4), compared to a 9.6% improvement last year.

Passage retrieval was actually detrimental. Combining passage-level and document-level evidence produced a 1.6% drop in average precision, as compared with last year's 15.7% improvement.

Combining InFinder query expansion and passage retrieval had little effect this year. There are many possible causes. The poor performance of passages alone is a likely cause. However, the new approach to using InFinder terms (i.e., not including them in the #passage operator) may also have been a factor.

The change to the estimation formula, described above, appears not to be the cause for the lower precision in this year's results. The new estimation formula provided a 4.7% increase in average precision for the basic query processing (QP, above) when compared to the old formula. Passages, InFinder, and a combination of the two all yielded slightly larger improvements with the old formula than with the new, but the improvements (2.1%, 4.5%, and 9.8%, respectively) were unimpressive. Further testing is required to isolate the cause.

6 Routing Results and Discussion

Two sets of results, INQ203 and INQ204, were evaluated in the document routing evaluation. Both sets of results were based on completely automatic processing of the TREC topic statement and relevance judgements into a query. The official evaluations are summarized below.

Recall	Precision (50 queries)						
	Query Processing (QP)	QP With InFinder (IF)		QP With Passage (PS)		QP and IF and PS (All)	
0	0.7248	0.7130	(−1.6)	0.6789	(−6.3)	0.7029	(−3.0)
10	0.5006	0.4863	(−2.9)	0.4778	(−4.6)	0.4929	(−1.5)
20	0.4074	0.4023	(−1.3)	0.3903	(−4.2)	0.3993	(−2.0)
30	0.3423	0.3419	(−0.1)	0.3316	(−3.1)	0.3484	(+1.8)
40	0.2775	0.2833	(+2.1)	0.2858	(+3.0)	0.2936	(+5.8)
50	0.2213	0.2380	(+7.5)	0.2320	(+4.8)	0.2475	(+11.8)
60	0.1608	0.1871	(+16.4)	0.1699	(+5.7)	0.1846	(+14.8)
70	0.0890	0.1126	(+26.5)	0.0853	(−4.2)	0.0933	(+4.8)
80	0.0484	0.0622	(+28.5)	0.0456	(−5.8)	0.0606	(+25.2)
90	0.0171	0.0246	(+43.9)	0.0232	(+35.7)	0.0276	(+61.4)
100	0.0012	0.0015	(+25.0)	0.0008	(−33.3)	0.0007	(−41.7)
avg	0.2330	0.2412	(+3.5)	0.2293	(−1.6)	0.2407	(+3.3)

Table 4: The effects of InFinder and passages on retrieval effectiveness.

Query Type	**Average Precision**			
	5 Docs	**30 Docs**	**100 Docs**	**11-Pt Avg**
INQ203	.65	.59	.48	.41
INQ204	.69 (+6.2%)	.58 (−1.8%)	.46 (−3.5%)	.40 (−2.9%)

The two methods of forming queries delivered very similar results. INQ204 was significantly better (6.2%) at the 5 document cutoff, but slightly worse (1–3%) at all other cutoffs and levels of recall. These differences, while consistent, are not considered noticeable or significant.

Recall that the difference between INQ203 and INQ204 is primarily in how the dynamic reweighting was performed. In INQ203, each concept was reweighted independently. In INQ204, they were reweighted in groups, depending upon their types. The small difference in effectiveness between the two experiments, suggests that either our dynamic reweighting had little effect, or that the "right" weight for a concept depends upon its type, i.e. whether it is a #1, #uw5, etc.

The concepts added to the query includes terms, pairs of adjacent terms, and pairs of terms occurring near each other. The following table demonstrates the relative value of each class of concepts. There are a few interesting results highlighted by that table. The large percentage improvements reflect the (by now unsurprising) value of relevance feedback. Note, though, that adding pairs of words which occur "nearby"—even as far as 50 words apart—has a more pronounced impact on effectiveness than adding single terms or adjacent pairs of terms.

The effect of adding pairs and terms is not cumulative, however, although using all of them does improve effectiveness more than any of them alone. Earlier experiments (not reported here) have suggested that the `#uw20` and `#uw50` pairs are each of roughly equal value, but that combining them provides virtually no increase in effectiveness over one of them alone. We are investigating methods for choosing the "best" proximity for a pair of terms found in relevant documents.

The last row of the table is precisely INQ204, and shows the value of Dynamic Feedback Optimization, which provided roughly a 3% gain over the original weights. There is some evidence to suggest that our method overfit, but the difference appears to be a matter of 3-4%.

	5 docs	30 docs	100 docs	11-pt
Original query (Q)	0.46	0.38	0.30	0.23
Q plus 50 terms	0.61	0.50	0.39	0.33 (+46%)
Q plus 50 #1's	0.53	0.47	0.36	0.29 (+26%)
Q plus 50 #uw5's	0.59	0.50	0.38	0.31 (+34%)
Q plus 50 #uw20's	0.66	0.55	0.43	0.35 (+54%)
Q plus 50 #uw50's	0.66	0.56	0.44	0.37 (+61%)
Q plus all	0.67	0.56	0.45	0.39 (+70%)
Q plus all, reweighted	0.69	0.58	0.46	0.40 (+75%)

Table 5: Effects of expanding by different concepts

7 Spanish Results and Discussion

Two sets of results, SIN010 and SIN011, were evaluated in the Spanish track. Both sets were based on automatic processing of TREC topics SP25-SP50 into queries, automatic query expansion, and use of document-level and passage-level evidence. SIN010 was evaluated with the normal INQUERY estimation formula and SIN011 was evaluated with a modified version of that formula. The official results for both query sets are summarized below.

Query Type	**Average Precision**			
	5 Docs	**30 Docs**	**100 Docs**	**11-Pt Avg**
SIN010	0.5040	0.4000	0.2804	0.2523
SIN011	0.5040 (+0.0%)	0.3880 (−3.0%)	0.2760 (−1.6%)	0.2458 (−2.6%)

Modification of the estimation formula did not improve performance. In fact it led to a slight decrease in performance overall (1-3%), but the drop is not significant.

Experiments were run after TREC-4 to investigate the effects of each phase of query processing. Results are given in Table 6. The average precision improves with query processing by 15.8% and 4.2% over raw words alone for SIN010 and SIN011, respectively. Although performance on raw words is 5.3% higher for SIN011 than for SIN010, the modified evaluation function used for SIN011 leads to a drop in average precision (2%-5%) with respect to SIN010 for each stage of query processing.

Passage retrieval led to the best performance. It yielded a 5% improvement in average precision for both SIN010 and SIN011. The modified evaluation function of the latter yielded lower average precision (5-7%) with passage-level and document-level query modification than did the original INQUERY evaluation function.

Query expansion with InFinder resulted in a drop in performance for both query sets (6-8%) and was worst at low levels of recall. This drop in performance is probably be due to low recall of noun phrases. Noun phrases were identified using a simple noun recognizer that only identifies roughly 55% of the nouns in any document (Section 3.2). As a result, InFinder may fail to consider many good noun phrases during indexing, and may also overestimate the importance of those noun phrases it does consider. Either would cause performance to drop. Tables 7 and 8 show the effects of query modification on performance for SIN010 and SIN011 respectively.

We are currently working on building an InFinder database using a POS tagger to identify nouns. The tagger should have much higher noun recall which is expected to improve results for InFinder query modification.

Recall	Precision (25 queries) SIN010			Precision (25 queries) SIN011		
	Raw Words (RW)	RW W/O Stop Phrases (NS)	NS WITH #PHRASE op (QP)	Raw Words (RW)	RW W/O Stop Phrases (NS)	NS WITH #PHRASE op (QP)
0	77.8	80.5 (+3.5%)	86.1 (+10.7%)	79.1	81.3 (+2.9%)	80.6 (+1.9%)
10	42.0	47.0 (+11.9%)	50.2 (+19.4%)	49.4	49.0 (−0.8%)	50.7 (+2.6%)
20	35.6	39.7 (+11.5%)	41.2 (+15.6%)	39.9	40.3 (+0.9%)	40.3 (+0.9%)
30	29.6	33.0 (+11.5%)	34.0 (+14.8%)	31.1	32.2 (+3.5%)	32.3 (+3.7%)
40	24.8	28.3 (+14.0%)	28.9 (+16.3%)	26.1	26.7 (+2.4%)	27.5 (+5.2%)
50	21.3	23.4 (+9.6%)	24.7 (+15.8%)	21.1	21.9 (+3.8%)	22.3 (+5.4%)
60	16.3	18.2 (+11.5%)	19.2 (+17.2%)	16.1	16.6 (+3.0%)	17.0 (+6.1%)
70	12.4	14.1 (+14.1%)	14.3 (+15.6%)	11.9	12.5 (+4.8%)	13.2 (+10.6%)
80	8.7	9.8 (+12.9%)	10.4 (+19.6%)	8.5	8.9 (+4.9%)	10.0 (+18.3%)
90	4.8	5.3 (+11.0%)	6.4 (+34.2%)	4.9	5.4 (+9.6%)	6.5 (+33.0%)
100	0.7	0.7 (+8.0%)	2.0 (+201.9%)	0.7	0.7 (−0.9%)	0.7 (+3.5%)
avg	24.9	27.3 (+9.5%)	28.8 (+15.8%)	26.3	26.9 (+2.3%)	27.4 (+4.2%)

Table 6: The effect of query processing on the retrieval effectiveness of SIN010 and SIN011.

Recall	Precision (25 queries)						
	Query Processing (QP)	QP With InFinder (IF)		QP With Passage (PS)		QP and IF and PS (All)	
0	86.1	69.3	(−19.5%)	87.1	(+1.1%)	78.9	(−8.3%)
10	50.2	47.3	(−5.7%)	54.6	(+8.8%)	49.2	(−1.9%)
20	41.2	39.4	(−4.3%)	42.7	(+3.8%)	42.2	(+2.6%)
30	34.0	34.6	(+1.8%)	35.3	(+3.8%)	35.1	(+3.2%)
40	28.9	28.8	(−0.4%)	31.5	(+9.1%)	30.6	(+6.0%)
50	24.7	23.4	(−5.3%)	26.2	(+5.9%)	25.4	(+3.0%)
60	19.2	18.8	(−1.7%)	19.4	(+1.5%)	21.0	(+9.5%)
70	14.3	15.6	(+9.4%)	14.9	(+4.3%)	15.7	(+9.8%)
80	10.4	11.2	(+7.9%)	11.1	(+7.4%)	11.6	(+11.8%)
90	6.4	7.9	(+22.8%)	6.9	(+8.2%)	7.6	(+19.5%)
100	2.0	2.4	(+20.7%)	2.1	(+2.5%)	2.3	(+12.3%)
avg	28.8	27.2	(−5.9%)	30.2	(+4.6%)	29.1	(+0.8%)

Table 7: The effects of passages and InFinder on the retrieval effectiveness of SIN010.

Recall	Precision (25 queries)						
	Query	QP With		QP With		QP and IF	
	Processing	InFinder		Passage		and PS	
	(QP)	(IF)		(PS)		(All)	
0	80.6	66.3	(−17.7%)	77.0	(−4.5%)	71.5	(−11.3%)
10	50.7	46.3	(−8.7%)	52.8	(+4.0%)	49.7	(−2.0%)
20	40.3	37.9	(−6.0%)	41.9	(+4.0%)	41.0	(+1.7%)
30	32.3	30.6	(−5.1%)	34.2	(+5.9%)	32.5	(+0.6%)
40	27.5	25.9	(−5.8%)	29.6	(+7.8%)	28.3	(+2.8%)
50	22.3	21.4	(−4.1%)	25.4	(+14.2%)	24.1	(+8.0%)
60	17.0	16.4	(−3.6%)	19.8	(+16.1%)	18.9	(+10.7%)
70	13.2	13.5	(+2.5%)	15.2	(+15.4%)	15.1	(+14.6%)
80	10.0	10.4	(+4.1%)	11.4	(+13.4%)	11.9	(+19.0%)
90	6.5	7.7	(+18.2%)	7.2	(+11.1%)	7.6	(+17.0%)
100	0.7	1.4	(+107.7%)	1.4	(+99.5%)	1.5	(+119.3%)
avg	27.4	25.3	(−7.7%)	28.7	(+4.9%)	27.5	(+0.3%)

Table 8: The effects of passages and InFinder on the retrieval effectiveness of SIN011.

8 Collection Merging Results and Discussion

Five sets of results, INQ205 – INQ209, were evaluated in the collection-merging document retrieval evaluation. After the results were submitted, we discovered that we inadvertently submitted the INQ206 results twice, as INQ206, and as INQ209. In the tables below, INQ209c is an unofficial run showing what we intended to submit (the "corrected" run). The official INQ209 results are not shown, because they are identical to INQ206.

The INQ201 adhoc run was treated as the baseline run. It gives the results of treating the subcollections as a single collection, the traditional IR paradigm. The official evaluations, and one unofficial run, are summarized below.

Query Type	**Average Precision**			
	5 Docs	**30 Docs**	**100 Docs**	**11-Pt Avg**
INQ201	.51	.35	.25	.24
INQ207	.49 (−4.8%)	.35 (−2.1%)	.24 (−3.8%)	.21 (−13.4%)
INQ208	.48 (−6.4%)	.35 (−0.8%)	.24 (−3.5%)	.21 (−13.4%)
INQ205	.50 (−2.4%)	.35 (−2.1%)	.24 (−6.2%)	.21 (−14.7%)
INQ206	.51 (−0.8%)	.35 (−1.4%)	.24 (−6.1%)	.21 (−15.0%)
INQ209c	.41 (−19.2%)	.33 (−6.9%)	.24 (−5.7%)	.18 (−26.8%)

The data support several conclusions.

- The "modified" term weighting functions were more effective than the "traditional" term weighting functions at very low recall (1-15 documents). After that, they were generally worse.

- Normalizing the document score based on the maximum and minimum scores that the query could generate in the collection was marginally useful with "traditional" term weighting, and more consistently useful with "modified" term weighting. However, the effects were small in both cases.

- Merging rankings based on document rank was significantly worse than merging based on document score. This is not a surprising result; indeed, the result is consistent with what others have found [13].

In general, we were pleased with the results. The best methods (INQ205 and INQ206) are not significantly worse than the single collection results until nearly 100 documents are retrieved. Even the worst method (INQ209) is probably adequate for interactive use.

9 Summary and Conclusions

We continue to believe in highly structured queries, sophisticated query processing, and in combining multiple sources of evidence. However, the adhoc experiments showed that we still have much to learn about these subjects. Query processing and structured queries were generally useful, but query expansion and passage-level evidence were not.

The routing track occupied more of our attention this year, which appears to have paid off. However, the relative similarity of our routing runs raises questions about the new dynamic reweighting algorithm. Much of the routing effectiveness may be due more to traditional factors, e.g., better term selection or using a wide range of proximity operators, than to the reweighting algorithm.

The Spanish track is a useful part of our research on foreign languages and IR. Last year we were happy to have Spanish results. This year, we wanted the same high level of effectiveness that we see in English. We expected the Spanish InFinder to help significantly, but it did not, suggesting that more research is required.

The collection merging experiments were a first step down the path to effective networked retrieval systems. The first step was surprisingly good, given the brevity and difficulty of the adhoc queries. However, there were so few collections that no strong conclusions can be drawn. We would like to see at least 100 (presumably smaller) collections in future evaluations.

Acknowledgements

We thank John Broglio, Stephen Harding, and Dan Nachbar for their assistance in the work described here. This paper is based on research supported by NRaD Contract Number N66001-94-D-6054 and the NSF Center for Intelligent Information Retrieval at the University of Massachusetts, Amherst.

References

[1] J. Broglio, W. B. Croft, J. Callan, and D. Nachbar. Document retrieval and routing using the INQUERY system. In D. Harman, editor, *Proceedings of the Third Text REtrieval Conference (TREC-3)*, pages 29–38. National Institute of Standards and Technology Special Publication 500-225, 1995.

[2] C. Buckley and G. Salton. Optimization of relevance feedback weights. In *Proceedings of the Eighteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 351–357, Seattle, 1995. Association for Computing Machinery.

[3] J. P. Callan. Passage-level evidence in document retrieval. In *Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 302–310, Dublin, Ireland, 1994. Association for Computing Machinery.

[4] J. P. Callan, W. B. Croft, and J. Broglio. TREC and TIPSTER experiments with INQUERY. *Information Processing and Management*, 31(3):327–343, 1995.

[5] J. P. Callan, W. B. Croft, and S. M. Harding. The INQUERY retrieval system. In *Proceedings of the Third International Conference on Database and Expert Systems Applications*, pages 78–83, Valencia, Spain, 1992. Springer-Verlag.

[6] J. P. Callan, Z. Lu, and W. B. Croft. Searching distributed collections with inference networks. In *Proceedings of the Eighteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 21–28, Seattle, 1995. Association for Computing Machinery.

[7] W. B. Croft, J. Callan, and J. Broglio. TREC-2 routing and ad-hoc retrieval evaluation using the INQUERY system. In D. Harman, editor, *Proceedings of the Second Text REtrieval Conference (TREC-2)*. National Institute of Standards and Technology Special Publication 500-215, 1994.

[8] Y. Jing and W. B. Croft. An association thesaurus for information retrieval. In *RIAO 94 Conference Proceedings*, pages 146–160, New York, October 1994.

[9] S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-3. In D. K. Harman, editor, *The Third Text REtrieval Conference (TREC-3)*, Gaithersburg, MD, (in press). National Institute of Standards and Technology, Special Publication 500-225.

[10] Richard V. Teschler. *Spanish Orthography, Morphology and Syntax for Bilingual Educators.* University Press of America, Inc., 1985.

[11] Howard Turtle and W. Bruce Croft. Inference networks for document retrieval. In Jean-Luc Vidick, editor, *Proceedings of the 13^{th} International Conference on Research and Development in Information Retrieval*, pages 1–24. ACM, September 1990.

[12] Howard R. Turtle and W. Bruce Croft. Efficient probabilistic inference for text retrieval. In *RIAO 3 Conference Proceedings*, pages 644–661, Barcelona, Spain, April 1991.

[13] E. M. Voorhees, N. K. Gupta, and B. Johnson-Laird. The collection fusion problem. In D. K. Harman, editor, *The Third Text REtrieval Conference (TREC-3)*, Gaithersburg, MD, (in press). National Institute of Standards and Technology, Special Publication 500-225.

[14] J. Xu, J. Broglio, and W. B. Croft. The design and implementation of a part of speech tagger for english. Technical Report IR-52, Center for Intelligent Information Retrieval, Department of Computer Science, University of Massachusetts, 1994.

Logistic Regression at TREC4: Probabilistic Retrieval from Full Text Document Collections

Fredric C. Gey, Aitao Chen, Jianzhang He and Jason Meggs
U.C. Data Archive & Technical Assistance (UC DATA)
University of California, Berkeley
e-mail: gey@ucdata.berkeley.edu

ABSTRACT

The Berkeley experiments for TREC4 extend those of TREC3 in three ways: for ad-hoc retrieval we retain the manual reformulations of the topics and experiment with limited query expansion based upon the assumption that top documents are relevant (this experiment was an interesting failure); for routing retrieval we introduce a logistic regression which assumes relevance weights to be only one clue among several in predicting probability of relevance. Finally, for Spanish retrieval we retrain the basic logistic regression equations to apply to the statistical distributions of Spanish words. In addition we apply two approaches to Spanish stemming, one which attempts to resolve verb variants into a standardized form, the other of which eschews stemming in favor of a massive stop word list of variants of common words.

1. Introduction and history of Berkeley TREC participation

For the past several years the UC Berekley Text Retrieval Research Group has been developing an approach to retrieving full-text documents from collections as large as 1 million documents, an approach which returns documents relevant to a query in order of estimated probability of relevance. Using the well-known technique of logistic regression, we predict the dichotomous variable 'relevance' as a function of statistical clues such as query term frequency in document and collection.

The Berkeley group began its participation with TREC1 ad-hoc track using the concept of staged logistic regression (Cooper, Gey, Chen 1993). In this formulation relevance was predicted as a logodds formula for query-document-term triples and statistical clues associated with these triples such as IDF, and term frequencies in document and query. Since the summation of these clues utilized the principle of linked dependence, a second stage regression was introduced between query-document pairs to compensate for known dependencies. The results in TREC1 were compromised by the inadequate training set and incomplete relevance judgments so necessary for adequate statistical fitting. In TREC1 Berkeley submitted no routing entry.

For TREC2 a single stage regression was introduced in the ad-hoc track which was based upon the concepts of "optimized relative frequencies" and the explicit use of number of match terms between document and query as a important statistical clue used in the logistic regression (Cooper Gey Chen 1994). For routing retrieval in TREC2, Berkeley used a logistic regression on relevance history terms weights for term presence or term absence from the document.

In the TREC3 ad-hoc development, Berkeley introduced a concept of averaging statistical clues across terms in common with query and document and performing a single regression against the averaged clues and the number of terms in common between document and query (Cooper, Chen, Gey 1995). The clues were carefully chosen to mirror the statistical weights of the tfidf vector space model. In addition for TREC3 Berkeley introduced the idea of manual reformulation and, where called-for, expansion of the TREC3 topics. Search terms were gathered by search of a similar but independent database, the University of California MELVYL news database. The use of this approach is based upon the idea that there exists some optimal version of a query which will produce the most relevant documents in the best order, and that human intervention based upon search experience can approximate the optimal query. For TREC3 routing, Berkeley used massive query expansion from relevant documents by computing a χ^2 relationship between terms and relevance as the discriminating factor in choice of terms. All terms which passed a five percent significance test for each query were added to the query. Original query terms which passed the test were weighted five times more than other terms of the expansion. This expansion produced from 300 to 4,114 terms, depending on the query.

2. TREC4 ad-hoc methodology

Our review and analysis of the TREC3 results showed some disappointment with the automatic run Brkly6 which used the formula of averaging clues across terms. The Brkly6 automatic run had an overall average precision of 0.2775 whereas the TREC2 fully automatic run Brkly3 had an overall average precision of 0.3270. Independent analysis (Fontaine, 1995) has shown that the Brkly6 algorithm performs significantly better than the vector space model. Although these performance measures were for different query sets and different collections, they were enough to make us reconsider whether the reformulation of TREC3 should be continued. We embarked on a series of experiments comparing the two algorithms which showed that the TREC2 Brkly6 algorithm performed about 10 percent better overall in a number of tests. Thus for TREC4 we made the decision to retreat to the TREC2 formula for our prediction of logodds of relevance between a query Q and document D. In particular the logodds of relevance when there are M match terms in common between query and document, we obtain a formula for M properties $A_1 \cdots A_M$, each associated with a match term, equation (1):

$$\log O(R \mid A_1, \ldots, A_M) \approx c_0 + c_1 f(M) \sum_1^M X_{m,1}$$

$$+ \cdots + c_K f(M) \sum_{m=1}^M X_{m,K}$$

$$+ c_{K+1} M + c_{K+2} M^2$$

where the function $f(M)$ is one that drops gently with increasing M, say $\frac{1}{\sqrt{M}}$ which exerts a damping influence as the number of match terms increases. Statistical tests showed that the M^2 clue did not make a statistically significant contribution and so it was dropped.

The properties chosen for the final equation were what were called in TREC2 "optimized relative frequencies". Thus the final equation after fitting was Equation (2):

$$\log O(R \mid A_1, \ldots, A_M) \approx$$

$$-3.51 + \frac{1}{\sqrt{M} + 1} \quad \Phi + 0.0929\, M$$

where Φ is the expression

$$37.4 \sum_{m=1}^M X_{m,1} + 0.330 \sum_{m=1}^M X_{m,2} + 0.1937 \sum_{m=1}^M X_{m,3}$$

Here

$X_{m,1}$ = number of times the m'th stem occurs in the query, divided by (total number of all stem occurrences in query + 35);

$X_{m,2}$ = number of times the m'th stem occurs in the document, divided by (total number of all stem occurrences in document + 80), quotient logged;

$X_{m,3}$ = number of times the m'th stem occurs in the collection, divided by the total number of all stem occurrences in the collection, quotient logged;

M = number of distinct stems common to both query and document.

Equation (2) above remains unchanged from TREC2. In particular, no refitting of logistic coefficients was attempted. However, the Berkeley entries for TREC-4 utilized limited query expansion, described below. For a more complete discussion of optimized relative frequencies, the reader is referred to (Cooper Gey Chen, 1994).

2.1. Manual query reformulation

In TREC3 the Berkeley group introduced a manual reformulation of the topics into new and different queries by having intermediaries search a parallel database, in this case the News database of the University of California's MELVYL electronic catalog. This approach proved so successful that we continued the process for TREC4. This seemed to be very promising when we observed the parsimony of expression in the TREC4 adhoc topic statements. As an example of such a reformulation, here is the original query for topic 241 on professional malfeasance:

> *Find examples of doctor and lawyer groups considering penalties against members of their professions for malfeasance and show the results of such investigations*

The average precision for Brkly9 entry on this query was 0.0120, retrieving only 16 out of 62 relevant documents. The manual reformulation of the same topic leads to the following query:

> *Find examples of doctor and lawyer groups considering penalties against members of their professions for malfeasance and show the results of such investigations doctor physician lawyer ama aba bar bar association license license suspended suspended disciplinary disciplinary action action disbar stealing sanction misconduct misconduct criminal physician license suspended nurse accuse case confidentiality mediations state licensing boards*

This reformulation (augmented by query expansion described below) allowed the Brkly10 run to achieve best average precision of 0.3199 in retrieving 56 of the 62 relevant documents.

Overall, manual reformulation nearly doubled performance from 0.1388 (Brkly9) to 0.2660 (Brkly10). The salient features of manual query development are *specificity* (adding query terms which are examples or instances of events being sought) and *generality* (adding synonymous or closely related terms which might expand the net of documents being ranked).

Finally, we experimented with Boolean filtering on a set of manually chosen negative terms, such that if a negative term occurred within a document, the document would not be retrieved. While this seemed to have great promise for certain queries limited to the United States in geographic scope, experiments showed it to be a two-edged sword such that it increased precision for some queries while reducing it for others.

2.2. A cautionary tale of ad-hoc query expansion

In TREC3 the entries from City University of London (Robertson, Walker, Jones, Hancock-Beualieu, 1995) and Cornell University (Buckley Salton Allan Singhal, 1995) utilized query expansion based upon the assumption that a trial run would produce an initial document ranking from which additional query terms could be harvested out of the top documents. City University's approach was to choose a limited number of expanded terms with a complex weighting formula, also from the top 30 documents of the initial probe. Cornell's approach was to choose the top 30 documents and find the 300 "best" terms and expand the query thusly using Roccio weights. Berkeley felt there was merit in these approaches and began to experiment with past collections to see if the methodology would work within the logistic regression environment.

Our experiments showed that using 30 documents and even 100 additional terms seemed to make performance deteriorate.

Table I: Brkly10 Ad-hoc Entry
Average Precision for document/term expansions

0.2945*	10 terms	20 terms	50 terms	100 terms
10 docs	0.2821	**0.2660****	0.2335	0.1995
20 docs	0.2646	0.2533	0.2263	0.1857
30 docs	0.2589	0.2477	0.2138	0.1885

*** - No expansion, ** - Brkly10 official entry**

Eventually we settled on expansion by twenty 'best' terms from the top 10 retrieved documents. This expansion was applied to both the original query and the manually reformulated queries. Thus the Brkly9 run is obtained by taking the original topic, performing a trial run, and expanding the queries by 20 terms taken from the top 10 ranked documents. A similar process was used for Brkly10, except that the seed query was the manually reformulated one.

Following receipt of the relevance judgments, Berkeley did a more systematic series of runs connected with query expansion. These runs, summarized in Table I, show that Berkeley would have been better served by making its official entry the run without expansion, which at 0.2945 was nearly 10 percent better than the 10 document/ 20 term expansion of the actual entry. Moreover, the same is true of the automatic entry, Brkly9; the official entry using query expansion had a precision of 0.1388. Without query expansion, the automatic methodology of equation (2) achieves average precision of 0.2001, more than 45 percent better than our official entry.

Our experiments show that query expansion must be done judiciously in terms of choice of number of documents and choice of number of terms. Even the top ten documents can produce several hundred additional terms to choose from. Berkeley used the following methodology to choose the top 20 terms from an initial ranking of the top ten documents using the original query -- term weights were calculated according to the following formula:

$$\sum_{i=1}^{10} DAF/NTD*log(NTD/DF)$$

where

DAF is the term frequency in document D_i,
$\sum(DAF)$ is the total number of occurrences of a stem in the NTD top ranked documents.
NTD is the number of top ranked documents (e.g. 10)
DF is the document frequency. (i.e. The number of documents among the NTD top ranked documents that contains the stem)

For our official entries, the twenty terms with the highest weights were added to the original query to form the expanded query.

2.3. Failure analysis

Both the Brkly9 original query and Brkly10 manual query modification stumbled on queries which specified events, actions, or scenarios which were restricted geographically to the United States. For example topic 205 (for which Brkly10 entry retrieved only 3 of 310 relevant documents)

> *What evidence is there of paramilitary activity in the US*

and our manual reformulation

> *What evidence is there of paramilitary activity in the US paramilitary paramilitary paramilitary militia militia militia group right wing armed guns hate anti-government extremists bomb shooting waco Koernke threat attack idaho michigan keystone montana Koernke Koernke sherwood terrorism*

retrieved numerous documents about military and liberation activities of the African

National Congress in South Africa. Similarly topic 206,

> *Prognosis/viability of a political third party in U.S.*

retrieved a number of documents about Venezuelan elections into the top ranks. It is clear that judicious use of a limited scope knowledge-base of geographic locations might considerably improve results of queries of this type.

3. TREC4 routing methodology

To prepare for TREC4 routing, the Berkeley group ran a number of retrospective experiments on the TREC3 collection. We were motivated by the City University approach of limited query expansion which was a marked contrast from the massive query expansion which Berkeley used in TREC3. We ran several experiments which utilized χ^2 tests and relevance history weights (the so-called "Robertson status value" ranking) for choosing terms. In both tests we found the Cornell approach of cutting off term expansion at 300 terms to be effective, however we could not achieve equal performance by utilizing a small number of additional terms of highest precision. In addition our experiments showed that relevance history ranking was better than χ^2 ranking for choice of the 300 expansion terms. This seems to be puzzlingly different from the Xerox-Parc results reported in SIGIR95 (Schutze, Hull, Pedersen 1995).

3.1. Logistic regression in routing

In an endeavor to improve on blind-faith query expansion, the Berkeley group noticed that the City University formulae for TREC3 made use of an additional clues of term frequency in document and query as well as document length. This seemed to indicate that a combination of relevance weighting with other clues similar to the ad-hoc methodology might actually improve performance. For TREC4 routing, Berkeley arrived at the following formula for logodds of relevance, Equation (3):

$$\log O(R \mid Q,D) = -0.2983+0.9309*X_1-0.001622*X_2$$

$$-0.000003059*X_3+0.05316*X_4+0.04613*X_5$$

$$+0.01573*X_6-0.005281*X_7+3.9488*X_8-0.06526*X_9$$

Where the individual clues are defined by

X_1 Query prior probability,
X_2 Number of unique stems in a document,
X_3 Document length (i.e. number of stems in a document),
X_4 Sum of present weight over match stems,
X_5 Sum of absent weight over query stems absent in document,
X_6 Sum of the log of the historical idf over match stems,
X_7 Sum of the log of the document term frequency over match stems,
X_8 Sum of the log of the relative document term frequency over match stems,
X_9 Number of match stems.

This logistic regression on multiple clues where relevance history weights are only one clue among many improved performance considerably in experiments on the TREC3 collections. Our baseline run for routing (Brkly11) consists of the original query terms ranked by the above formula. The Brkly12 run uses the same formula, applied to the top 300 terms for each query chosen from past relevant documents by their relevance weight.

3.2. TREC4 routing performance

The results for TREC4 run Brkly11 are abysmal, with average precision of 0.0345, showing that using only the original query is completely inadequate for the routing retrieval problem. The Brkly12 run which did query expansion by 300 terms chosen in order of relevance history weight improves performance by a factor of seven to 0.2163. However, even this performance is below the median for most queries, puzzlingly low from our point of view, especially since the performance on query 17 ("Measures to control agrochemicals") was

0.8509, the best for that query.

Since receiving the results, we have rerun using the TREC-3 routing formulation with a χ^2 computation to choose the top 1000 expansion terms. This run achieved an average precision of 0.2675 approximately twenty percent better than our official entry.

4. TREC4 Spanish retrieval

In approaching Spanish retrieval the Berkeley group was first faced with the problem of developing a Spanish stemmer. There were two conflicting approaches to stemming of TREC3: Cornell's assertion that only extremely primitive stemming is needed, and that a stop-word list can be developed from the most frequent stems, manually reviewed to choose which stems should be kept, versus Univerity of Massachusetts assertion that sophisticated stemming produces a significant improvement. Mindful that it might not improve performance, Berkeley nevertheless undertook to develop a Spanish stemmer.

4.1. Spanish stemming software

Spanish stemming methodology follows some rules for morphology found in recent computational linguistic literature (Moreno and Goni, 1995). The Berkeley Spanish stemmer begins with removal of longest ending found, if any. If an ending was found, the software converts final c to z on stem then normalizes verb forms found into their normal form. Verbs are the most complex items to stem; they consist of regular, ortho-irregular, and irregular verbs of three classes, -ir, -er and -ar. The classification is made by the ending of the infinitive form of the verb, which is also the name of the verb.

The *regular verbs* have standard endings similar to Spanish nouns, and their stem does not change throughout the conjugation. It is relatively trivial to stem these verbs. *Ortho-irregular verbs* behave like regular verbs in most respects except for the fact that their stem changes throughout the conjugation into one or more new forms. These "stem-changing" or "radical" verbs fortunately follow fairly basic rules which are outlined in standard grammars of Spanish. It was and still is a matter of interest how ambiguous it is to back-track in that respelling. *Irregular verbs* are verbs that may partially follow regular or ortho-irregular patterns, save for one or more forms are totally unpredictable. No rule can be written for a change in a verb from, say, "ser" to "fui". There are unfortunately around 2,750 or more irregular verbs in the Spanish language. The task of normalizing irregular verbs would have been insurmountable had we not found a researcher at the University of Waterloo who was willing to provide us with his laboriously constructed machine-readable file of all verb forms for most irregular verbs (German, 1995). *Non-verbs: nouns, adverbs, adjectives* are generally much easier, since they rarely undergo respelling. They can have occasional irregularities such as the same form for plural as singular, or never assume a singular form (English example: what's the singular of pants?). This is still fairly straightforward unless extreme disambiguation is desired. The basic endings are found in any complete grammar, and nouns, adjectives and adverbs usually resolve to the same stem when they have the same basic meaning.

We also performed two forms of stemming, regular stemming which resolved word variants (mostly verb variants) into their normal form, and no collection stemming, but the construction of a massive stop word list by taking all words of a small stop list (the translation of the SMART list into Spanish) and expanding them into all possible variants. In the latter case we obtained a stop word list of some 10,000 words.

4.2. Logistic regression for Spanish

The Berkeley group methodology for retrieving Spanish was to utilize the proven formula for adhoc logistic regression and retrain on the Spanish training set (Spanish queries 1-25). Doing this resulted in the following Equation (4):

$$\log O(R \mid A_1, \ldots, A_M) \approx$$

$$-3.439 + \frac{1}{\sqrt{M} + 1} \Phi - 0.0143\, M$$

where Φ is the expression

$$84.86 \sum_{m=1}^{M} X_{m,1} + 0.508 \sum_{m=1}^{M} X_{m,2} - 0.3197 \sum_{m=1}^{M} X_{m,3}$$

where the $X_{m,k}$ are the same clues as in Equation (2) above.

4.3. Spanish entries -- manual reformulation

Mindful of the success of manual reformulation of English ad-hoc queries, we undertook to create manual versions of queries 25-50. Queries 1-25 were considered sufficiently rich in natural language expression as to make manual construction unnecessary. The process was to translate the queries into English, search the MELVYL NEWS database in English, then reformulate manual queries in English and translate them into Spanish.

4.4. Spanish results

The TREC4 official entry had an average precision of 0.1683, about equal to a number of other Spanish entries and considerably below the University of Central Florida entry. Manual reformulation in this case showed no improvement, with average precision of 0.1637. Following submission of the official entry, we discovered an error -- in computing the X_3 component of the regression, we used the collection size from the English collection (140 million stems) rather than the Spanish collection (14 million stems). Rerunning with the correct collection size showed insignificant change (average precision 0.1702) in the result. The most important run was that without stemming and the massively expanded stop word list, whose precision was 0.1009. Our conclusion is that stemming works for Spanish as well as English.

5. Acknowledgments

All Berkeley group research is based upon a modified version of the Cornell SMART system. Berkeley's modification to this software consists of replacement of the vector space similarity computation algorithms with probabilistic algorithms, together with such modifications of the SMART vector and inverted file data structures as necessary to incorporate additional clues. Support for this project came from UC DATA's research and development budget, augmented by research assistant funds from the UC Berkeley School of Information Management and Systems. We also acknowledge the encouragement and moral support of Professor William S. Cooper, whose basic ideas for the past half decade form the core of all our methods.

The generosity of Daniel German of the University of Waterloo in providing his Spanish irregular verb conjugation list made a crucial difference in our ability to deliver probabilistic retrieval of Spanish text.

6. References

Buckley, Chris, Gerard Salton, James Allan, Amit Singhal (1995), "Automatic Query Expansion using SMART: TREC 3," *Proceedings of the Third NIST Text Retrieval Conference (TREC3),* National Institute for Standards and Technology, Washington, DC, November 2-4, 1994, NIST Special Publication 500-225, April 1995, pp. 69-80.

Cooper, W. S.; Gey, F. C.; Chen, A. (1993). "Probabilistic retrieval in the TIPSTER Collections: An Application of Staged Logistic Regression." *The First Text REtrieval Conference (TREC-1).* NIST Special Publication 500-207. Washington, 73-88.

Cooper, W. S.; Gey, F. C.; Chen, A. (1994). "Full Text Retrieval based on Probabilistic Equations with Coefficients fitted by Logistic Regression." *The Second Text REtrieval Conference (TREC-2).* Washington, DC, August 31-September 2, 1993, NIST Special Publication 500-215. March 1994, pp.57-66.

Cooper, William S., Aitao Chen, and Fredric Gey (1995), "Experiments in Probabilistic

Retrieval of Full Text Documents," *Proceedings of the Third NIST Text Retrieval Conference (TREC3),* National Institute for Standards and Technology, Washington, DC, November 2-4, 1994, NIST Special Publication 500-225, April 1995, pp. 127-134.

Fontaine, Anne (1995), "Subelement indexing and Probabilistic Retrieval in the POSTGRES Database System", Masters Thesis, Computer Science Department, University of California, Berkeley, May 1995.

German, Daniel M., Private Communication, June 1995.

Moreno, A. and Goni, J (1995), "GRAMPAL: A Morphological Processor for Spanish implemented in Prolog", Technical Report, Dep. de Linguistica, Universidad Autonoma de Madrid, 1995.

Robertson, S.E., S. Walker, S. Jones, M.M. Hancock-Beualieu and M Gatfort (1995), "Okapi at TREC-3," *Proceedings of the Third NIST Text Retrieval Conference (TREC3),* National Institute for Standards and Technology, Washington, DC, November 2-4, 1994, NIST Special Publication 500-225, April 1995, pp. 109-126.

Schutze, Hinrich, David A. Hull, and Jan O. Pedersen (1995), "A Comparison of Classifiers and Document Representations," *Proceedings of SIGIR95, the 18th Annual Conference on Research and Development in Information retrieval,* Seattle Washington, July 9-12, 1995, pp. 229-237.

Okapi at TREC–4

S E Robertson S Walker M M Beaulieu M Gatford
A Payne

Centre for Interactive Systems Research
Department of Information Science
City University
Northampton Square
London EC1V 0HB
UK

Advisers: E Michael Keen (University of Wales, Aberystwyth), Karen Sparck Jones (Cambridge University), Peter Willett (University of Sheffield)

1 Introduction

Okapi at TRECs 2 and 3

During TRECs 2 and 3

- the new term-weighting functions were developed and refined as described in [1, Section 3.2]
- a method of runtime passage determination and searching was devised [1, Section 4];
- an inefficient but surprisingly effective way of choosing routing terms was developed [1, Section 6]);
- and a reasonably effective way of automatically expanding ad hoc queries with terms from documents retrieved in a pilot search was developed [1, Section 5].

TREC–4

City have submitted interactive runs in all the previous TRECs, with fairly undistinguished results. This time the main emphasis has been on the development of an entirely new interactive ad hoc search system (Sections 3 and Appendix). On the non-interactive side

routing term selection: there has been further work on methods of selecting routing terms;

manual and automatic ad hoc: the automatic ad hoc was done in more or less the same way as for TREC–3, but in view of the very brief topic statements a few runs were also done with manually edited queries.

2 The system

2.1 The Okapi Basic Search System (BSS)

The BSS, which has been used in all City's TREC experiments, is a set-oriented ranked output system designed primarily for probabilistic–type retrieval of textual material using inverted indexes. There is a family of built-in weighting functions as described in [1, Section 3]. In addition to weighting and ranking facilities it has the usual boolean and quasi-boolean (positional) operations. There are a number of non-standard set operations, some of which were added during the TREC–4 experiments.

- Varieties of AND and NOT operators intended for "limiting": these are used to reduce a ranked set to the subset implied by the operator and the right-hand operand(s), without affecting the weights of the elements of the left-hand operand.
- A unary operator which produces a set from the top-ranking N elements of its operand (example below).

```
FIND SET=<setnum> TOP=<num>
```

- A binary NOT–type operator which, when the right-hand operand R is a constituent of the (ranked) left-hand operand L, produces a new set which is what L would be if R had not been a constituent. This was used in routing term selection to speed up testing the effect of removing a term from a query.

Some enhancements were made to the data processing software. Indexes were simplified, and, to accommodate a single index to the million-plus documents of disks 1–3, redesigned so that an index may be distributed over a number of Unix filesystems.

Weighting functions

The functions available were identical to those used in TREC–3. The only ones used during TREC–4 were varieties of the **BM25** function [1, Section 3.2]

$$\sum_{T \in Q} w^{(1)} \frac{(k_1+1)tf}{K+tf} \frac{(k_3+1)qtf}{k_3+qtf} + k_2.|Q|.\frac{avdl-dl}{avdl+dl} \quad (1)$$

where

Q is a query, containing terms T

$w^{(1)}$ is the Robertson–Sparck Jones weight [3]

$$\log \frac{(r+0.5)/(R-r+0.5)}{(n-r+0.5)/(N-n-R+r+0.5)}$$

of T in Q

N is the number of items (documents) in the collection

n is the number of documents containing the term

R is the number of documents known to be relevant to a specific topic

r is the number of relevant documents containing the term

K is $k_1((1-b)+b.dl/avdl)$

k_1, b, k_2 and k_3 are parameters which depend on the database and possibly on the nature of the topics. For the TREC–4 experiments, typical values of k_1, k_3 and b were 1.0–2.0, 8 and 0.6–0.75 resp., and k_2 was zero throughout.

tf is the frequency of occurrence of the term within a specific document

qtf is the frequency of the term within the topic from which Q was derived

dl is the document length (arbitrary units)

$avdl$ is the average document length.

Passage determination and searching

The same technique was used as for TREC–3 [1, Section 4]. The theoretical minimum passage length was one algorithmically determined paragraph, paragraph length and position information being recorded in an auxiliary paragraph file for each database. Passages consisted of an integral number of paragraphs. The run-time arguments are

p_unit: the minimum number of paragraphs in a passage; passage lengths are a multiple of this, unless they hit the end of the document without reaching a multiple

p_step: the number of paragraphs to advance between passages

p_maxlen: the maximum number of paragraphs in a passage.

For example, if $p\_unit = 4$, $p\_step = 2$ and $p\_maxlen = 8$ and a document is 11 paragraphs long, the following passages are examined: 1–4, 1–8, 1–11, 3–6, 3–10, 3–11, 5–8, 5–11, 7–10, 7–11 and 9–11; if $p\_unit = p\_step = 1$ and $p\_maxlen = \infty$ every run of consecutive paragraphs is examined. It is also possible to set a *passage_avedoclen*, typically somewhat lower than the database's true average document length. In the tables, passage runs are sometimes notated in the form $(\langle p\_unit\rangle, \langle p\_step\rangle, \langle p\_maxlen\rangle[, \langle passage\_avedoclen\rangle])$.

Whatever the passage–argument values the entire document was also treated as a passage. The output from a passage search gave for each document a weight for the entire document and the location and weight of the highest–scoring passage. Obviously, passage searching is computationally demanding for long documents, with time complexity increasing with the square, or cube (if *p_maxlen* is infinite), of the document length, so it is essential to keep the number of documents examined as small as possible. This is achieved by performing passage searching in two stages. In the initial search whole documents only are examined, and the top ten thousand or so output to form an intermediate set, which it is hoped will contain most of the documents which would be found by a 1000–cutoff passage search. In the final stage passage searching is done on the intermediate set only, thus reducing the number of documents examined from perhaps a few hundred thousand to ten thousand.

Passage searching was used in all the official City runs, and usually gave an improvement in average precision of about 2–5 percent, sometimes at a cost in low precision. Most of the runs were done with passage arguments $(1,1,20)$ or $(4,2,32)$. There is little difference between the results, and the latter is quicker. In the interactive system the passage information enabled the interface to point the user to the "best" part of a long document.

2.2 Hardware

A single-processor Sun SS10 with 256 MB of core and about 25 GB of secondary storage was used as the main development machine and file server. Batch processing was also done on three other Suns, an IPX with 48 MB, a 4/330 with 40 MB and an IPC with almost no memory.

2.3 Databases

Three databases were used: disks 1, 2 & 3 for routing trials and some of the ad hoc development, disks 2 & 3 for automatic, manual and interactive ad hoc, and "disk 4" for the predictive routing runs. The same three–field structure was used, common to all source datasets, as for TREC–3. The first field was always the DOCNO and the third field contained all the searchable text, mainly the TEXT portions but also headline or title-like material for some datasets and documents. The second field was unindexed (and unsearchable) and so only (possibly) useful for display to users of the interactive system. It was empty except in the case of SJM, when it contained the DESCRIPT field; and the Ziff JOURNAL, AUTHOR and DESCRIPTORS fields. All the databases were set up in such a way that search-time passage determination and searching could be done.

3 Interactive track experiment

For TREC–4 the Okapi team concentrated more on the interactive track. The object was to build on our previous experience in both automatic and interactive methods and optimize on a combined approach. The focus was on two aspects: the development of an interface more amenable for interactive query formulation, and on passage retrieval for relevance feedback. The searchers were members of the team who were experienced with the system and played the role of an intermediary searching on behalf of a remote user. A post-search questionnaire was administered in order to obtain more qualitative data on the searchers' perception of the query, the effort required in selecting terms and making relevance judgments, and overall search satisfaction.

The Appendix includes a description of the interactive system, a short summary report on the search process, including a search narrative for query 236 on areas of disagreement on the laws of the sea, as well as tabular results of the questionnaire and the guidelines given to the searchers (E).

A major concern in carrying out the interactive experiment was to ensure that the task could be carried out in the time allocated with minimal loading on the user, whilst at the same time exploiting the system's functionality to the maximum. Unlike the command-based expert interface used in TREC–3, the current GUI interface made full use of direct manipulation interaction for making selections and executing commands and for the concurrent display of different information. The main features and improvements made it easier for the searcher to keep track of the search as it developed and included the following:

- the ability to enter search terms as phrases (interpreted by the system as a combination of an adjacency search and a lower–weight same-sentence search—see Appendix A.2.1);
- the use of a single dynamically ranked and updated current query term set, including query expansion terms (single words) extracted from relevant documents, with interactive addition and removal of terms by the searcher;
- the presentation of a more informative hitlist with short document titles or equivalent, together with source, length and query term occurrence information;
- the direct viewing of highlighted best passages of documents to enable the searcher to make relevance judgments more quickly and also to choose between the full document and best passage only for relevance feedback;
- the display of user selected relevant items in ranked order in a separate window, updating the searcher on the current state of the search results.

A search session could thus be broken down into one or more iterations of three basic sequential activities, each of which was handled by a separate window: entering and manipulating query terms, viewing hitlists of titles and displaying documents to make relevance judgments.

3.1 Generating initial query terms

Since the topics provided so little context, searchers were asked how difficult it was to interpret the topics. Just over half (13) of the topics were found to be straightforward and easy to ascertain what was required. Searchers expressed reservations on nine of the topics, which they described as moderately easy. For the most part they anticipated some difficulty in finding a range of documents to reflect all aspects of the topic. Three queries (205, 209, 211) were deemed to be difficult to formulate because the searcher did not know what sort of documents to expect.

The short topics undoubtedly affected the number of terms used for the initial query formulation. An average of 4.7 terms was used, compared with 8.1 in the interactive routing task for TREC–3. Terms used divided roughly equally between phrases and single words (2.3 and 2.4 respectively). Searchers declared that it was easy or moderately easy to generate initial query terms for 18 of the topics. In the majority of cases this involved both extracting terms directly from the topic and adding other terms. For the six who found it difficult, or in one case very difficult (topic 236), the problem was thinking up synonyms or appropriate words to describe the different aspects of the topic or to express abstract ideas. For three out of the seven queries which were described as difficult or very difficult, only a single term was entered to start the search and the rest had two, three, six and six terms respectively. For the seven queries described as easy, the number of terms ranged from three to nine.

3.2 Expanded query term lists: adding and removing terms

Terms were added to the query during the search either by being entered directly, or as a result of expansion by the system from marked documents. Any terms could be removed by the searchers.

An average of 5.2 terms were added directly by searchers after the initial search and the range was 0–19, with no terms added in six searches. Searchers who added new terms indicated that for 14 of the searches they did so to improve recall, and in only four of the searches was the purpose to shift the emphasis or improve precision. Similarly, searchers seemed to remove terms from term sets primarily to improve recall. In 16 searches proper names were removed and in eight numbers were removed. Terms deemed as inconsequential or irrelevant were discarded in seven searches, whereas in two searches terms were also removed if they appeared both in phrases and as single terms. In only one case a searcher divulged that terms were removed in an attempt to focus the search.

An average of 39.3 terms were removed in the course of a search and the range was 0–113. This far exceeded the number of terms removed in the interactive routing experiment in TREC–3, where the mean was 3.2 and the range 0–15. In the current round a more extensive list of extracted terms was displayed when searchers used the query expansion facility. This list extended beyond the cut-off used by the system when performing a search, in other words, the displayed list included terms lower down the ranking, which would not be used in the search unless brought up by searcher action. Clearly it was easier for users to discard terms suggested by the system than to generate their own. In some instances searchers did find relevant terms in documents and added them to the query term set themselves. Overall searchers depended heavily on the query expansion facility to formulate their queries.

3.3 Query expansion and final query terms

Searchers arrived at a final query formulation normally after an average of 3.6 iterations. An iteration here was defined as the execution of the search command, which may or may not have been preceded by the use of the query expansion facility. Query expansion was used only once in 20 of the searches, twice in two searches and not at all in three. For 11 searches where expansion took place, searchers indicated that they had expanded because they had found relevant documents but had exhausted the current list. Their intention was to get more items similar to what had already been retrieved. In five cases the intention was to find documents which were different from those already found to be relevant and in six searches query expansion was used out of desperation because not enough relevant documents could be found.

The length of the topics had little impact on the number of terms in the final query where the mean was 16.9 and 16.6 for TREC–3 and TREC–4 respectively. The main explanation for this is simply that for interactive searching the system used the 20 top ranking terms for query expansion and the searcher had the option to remove some. Out of the average number of 16.6 terms, 12.8 were individual terms and 3.8 were phrases. Overall the system retained 121 out of the total of 133 phrases generated by the searchers.

Searchers did not find it difficult to determine what terms should constitute the offline query for the secondary task. For 21 searches they found it easy or moderately easy and for only four they found it difficult or very difficult. In thirteen searches the final interactive query formulation was altered for the secondary offline stage of the interactive task. In 11 of the searches, terms were both added and removed, whereas in the remaining two searches terms were added or removed only.

3.4 Viewing hitlists of titles

In just over half of the searches one or two hitlists of ranked items were displayed, and between three and seven hitlists in the remainder. On average searchers browsed through 54.2 titles per search. Searchers were asked how easy it was to find relevant items in the initial hitlist and in 13 instances they found it easy or moderately easy compared to 12 where they found it difficult or very difficult. There appeared to be a strong correlation between users' assessment of the degree of ease in generating initial query terms and their perception of the quality of the results in the initial hitlist. For example it turned out to be difficult or very difficult to identify possible relevant items from the hitlist for five out of the six searches where searchers experienced problems in generating appropriate query terms in the first place.

3.5 Passage retrieval, relevance judgments and relevance feedback

For the interactive experiment passage retrieval was applied after the initial retrieval of the document set and served to re-rank the documents according to best passage. Records were displayed with the best passage and query terms both highlighted. In some cases the weighting operation did not identify a best passage [1] and records were displayed with query terms highlighted only. In the case of very long documents (more than 10K), only the best passage was displayed.

The purpose of passage retrieval in interactive searching was to:

- improve the ranking of documents
- enhance the display and viewing of documents
- assist users in making relevance judgments
- improve term selection for query expansion.

By highlighting the section of the document where query terms were the most concentrated, the searcher could use this as a starting point. This was particularly helpful for long documents which might have required more extensive scanning. In the case of documents dealing with several topics, best passage retrieval made it possible to go straight to the section most appropriate to the query.

In making relevance judgments, searchers were in effect not only indicating the relevance of the document for the query but they were also choosing between the whole document or the passage as suggested by the system for extracting terms for query expansion. This was particularly useful in the case of multi-topic documents. For 64% of positive relevance judgments, the searcher indicated that the system–specified best passage was relevant (this implied that *only* this passage was used for query expansion). On average searchers made 17 positive relevance judgments per search as opposed to 14 negative ones. Diagnostics on the effectiveness of passage retrieval for interactive searching have yet to be carried out to compare the ranking of documents, the degree of agreement between the system and the user for term selection as well as the retrieval effectiveness of using whole documents or best passages only for query expansion.

3.6 User satisfaction

An attempt was made to get some measure of user satisfaction by asking searchers how they rated the overall difficulty of the topics and how they perceived the success of their searches. Fifteen topics were rated as easy or moderately easy and ten as difficult or very difficult. Searches were classed as problematic when searchers found it difficult to express the search more precisely and to focus on a particular aspect.

With respect to how successful they considered their searches, nine were considered to be successful, another nine moderately successful and seven not successful. Success seemed to be expressed here both in terms of recall and precision. Unsuccessful searches seemed to be those where few items were found and the searcher felt that perhaps there should have been more there. In the case of moderately successful searches, these seemed to be searches where the relevant items covered some aspects of the topic but not all. Six out of the seven searches categorized as unsuccessful were in fact deemed to have produced poor results in the initial hitlist. Of the remaining six searches which also produced poor initial results, four turned out to be moderately successful and two successful.

3.7 Results of the interactive searches

For the primary task (Table 8 in the Appendix), the macro–average recall and precision figures are 15% and 66% respectively. An average of 0.38 relevant documents were found per minute of elapsed search time.

Further measures were also calculated based on different time intervals and the number of iterations per search sessions (Tables 9 and 10 in the Appendix). Taking into account the end point results in these tables, searchers saw on average 31 full documents per session (column K), that is one per minute, and made positive relevant judgments for just over half of these (17, column F). Of the 14.5 average number of official relevant documents seen by the searcher (column L), 3 (column M) or 24% (column N) were rejected. Figure 1a shows that as more documents were seen over time in a search session, more positive relevance judgments were made, an indication that the search formulation improved. Also of those items selected by the searcher (Figure 1c, H), the proportion officially judged relevant improved in the course of the search, showing that searchers became more confident and familiar with the topic. This improvement happens in the first 10 minutes.

The same measures calculated at each iteration demonstrate the same trend. An iteration is defined here as a submission of a new search statement. The flattening effect in Figures 1b and 1d is partly due to the fact that not all the searches reached the maximum number of iterations. This does not occur in the case of the duration of search sessions because no session ended before the penultimate 20 minutes point. However, the flattening effect on precision appears to be genuine (Figure 1c). Clearly the initial queries are not very good but improve as they are reformulated.

For the secondary task, the results are given in Table 1 below, together with our automatic and manual ad hoc results, as well as a diagnostic run. The diagnostic run was based on the final query formulation without the assumption that some documents had been retrieved (i.e. without frozen ranks). The very high precision at 5 documents reflects that this is an "optimal" formulation, whereas the results of the actual interactive session

[1] i.e. the best passage was in fact the whole document.

Table 1: Secondary task evaluation summary compared with City non-interactive ad hoc results

Run	AveP	P5	P30	P100	R-Prec	Rcl
Cityi1	0.274	0.520	0.456	0.321	0.340	0.543
Citya1	0.243	0.480	0.353	0.235	0.279	0.531
Citym1	0.235	0.464	0.335	0.233	0.272	0.531
Citydiag	0.241	0.592	0.425	0.285	0.285	0.530

reflect searchers' initial uncertainty. The performance of the diagnostic run seems to be worse at high document ranks, so that the average precision comes down to more or less the same as the automatic ad hoc results (citya1).

4 Automatic routing

The task involves selecting terms, assigning weights to them and combining the terms in a suitable way, while hoping that whatever training datasets are used are not too different from the test data. One could in theory take all the index terms in some training set and perform some multidimensional combinatorial feat to arrive at an approach to an optimal set of (term, weight) pairs. In practice, none of us has enough computing resources even to reach some state of sub-optimality; enough constraints have to be added to reduce the task to something which can be done in a matter of a few weeks.

The constraints used were as follows:

1. terms were restricted to those present in officially relevant documents;

2. terms were assigned $w^{(1)}$ weights (Section 2.1) in accordance with their occurrence in the relevant documents;

3. terms were arranged in descending order of their Robertson Selection Value (*RSV*) [4];

4. the number of terms considered by the second–level selection process was limited to a fixed number, never greater than 200;

5. terms were added to or removed from the query singly, so that only a minute proportion of the possible combinations of terms were considered.

This type of procedure was first used by City for TREC–3, with encouraging results; for TREC–1 and TREC–2 [5] first–level term selection was as above (items 1–3), but there was no second–level selection—either a fixed number of terms was used working down the list, or the number of terms used depended on the topic.

4.1 Term extraction

Three sets of potential query terms were produced, one from the full disks 1, 2 & 3 database (*full*) and one each from its odd and even half-collection sub-databases (*odd* and *even* resp). Every non-stop term was extracted and $w^{(1)}$ weights, *RSV*s and other statistics recorded. Those with *RSV* less than 3 were discarded.

4.2 Scoring functions for term selection

In TREC–3 the second–level term selection score was in most cases the average precision (using cutoff at 1000 documents) from the official TREC evaluation procedure. At every stage a new search process was invoked, the IDs of the top 1000 documents output, and the TREC evaluation run in the usual way. This rendered the process so slow that a number of constraints were used simply to enable the selection to run acceptably fast: the total number of terms was limited to 20, no term was reconsidered if it had failed to increase the score at some previous stage, the process stopped if eight successive terms failed and, finally, an absolute time limit was applied.

For TREC–4 a number of more rapidly computable measures were tried.

BROOKES The "Brookes measure" [2], is a normalized form of the difference in mean weights between relevant and non-relevant documents in the retrieved set (see equation 2).

$$\frac{\mu_{rel} - \mu_{nonrel}}{\sqrt{(\sigma^2_{rel} + \sigma^2_{nonrel})}} \qquad (2)$$

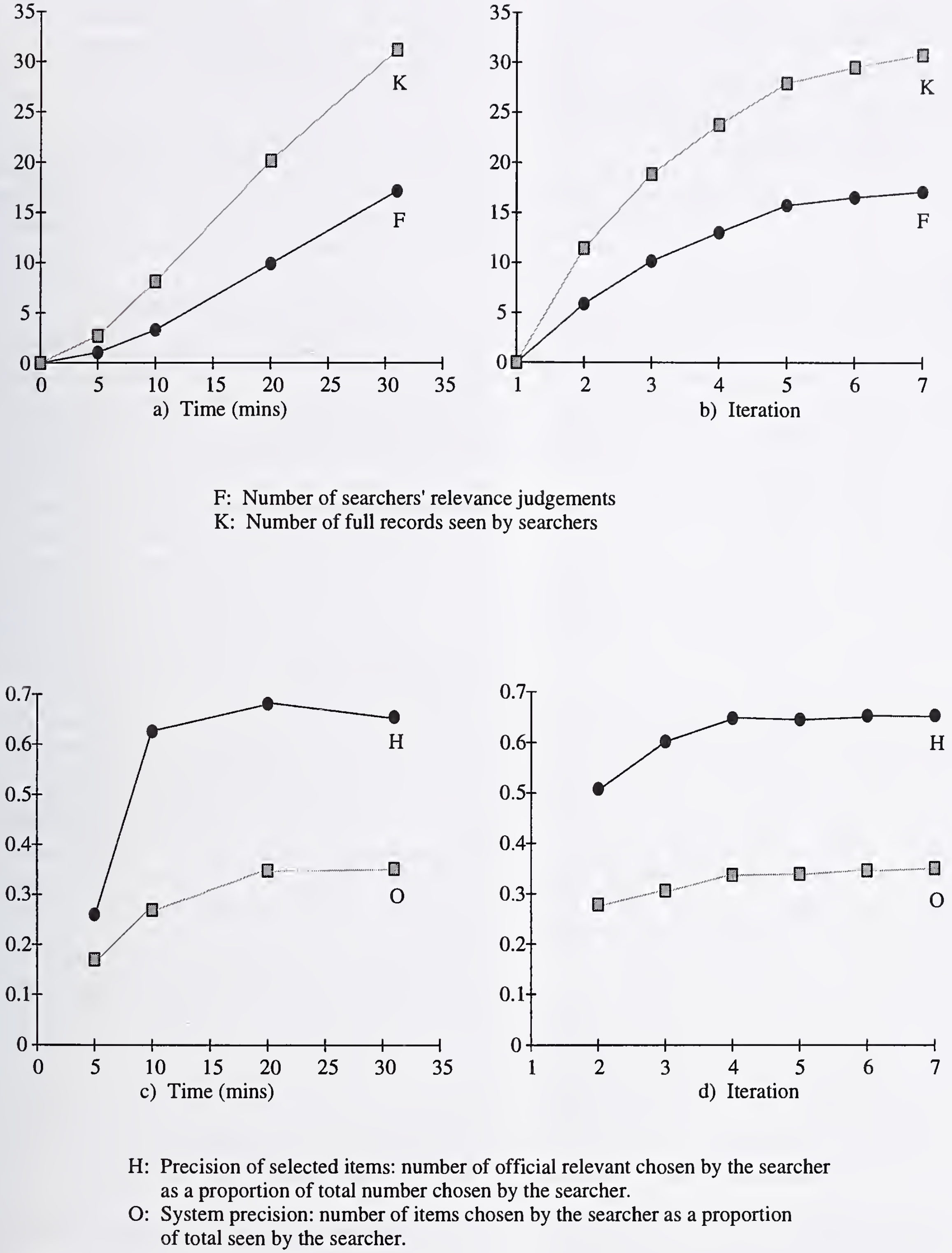
This was efficiently evaluated by calculating the sum and sum of squares of document scores on the fly within the BSS set combination routines. The Brookes measure, although intuitively appealing, did not prove to correlate well with the TREC average precision measure.

DIFFM Un-normalized difference of means between relevant and non-relevant documents. This gave results which were very similar to those from the Brookes measure.

3–PT A 3–point precision average—the mean of precisions at $2R$, R and $R/2$.

RPREC The precision after R documents have been retrieved.

AVEP A multi-point precision average on the top D documents, usually calculated on $D = 1000$ and at D/g, $2D/g$, …, D, where the granularity parameter g was varied between 3 and 20. This gave the best results, noticeably better than our TREC–3 results using TREC evaluation average precision



F: Number of searchers' relevance judgements
K: Number of full records seen by searchers

H: Precision of selected items: number of official relevant chosen by the searcher as a proportion of total number chosen by the searcher.
O: System precision: number of items chosen by the searcher as a proportion of total seen by the searcher.

Figure 1: Searcher relevance judgments, Documents shown against Time, Iteration boundaries

under restrictive conditions on the number of terms tried, etc.

TREC–AVEP Average precision identical to that produced by the official TREC evaluation program.

CBM1 This is a linear combination of *TREC–AVEP* and *BROOKES* with *TREC–AVEP* being given ten times the weight of *BROOKES* since their mean values are roughly in this ratio.

The effect of varying scoring function is shown in Table 2

4.3 Term selection algorithms

For all selection methods, potential terms were first arranged in descending *RSV* order to form a term-set $\mathcal{T}$ containing a predetermined maximum number T of terms. For almost all runs terms which occurred more than once in the topic statement were given a bonus by using a k_3 value of 8 in the weighting formula 1. When an iteration starts there is assumed to be a set of already selected terms (possibly empty) forming the current query $\mathcal{Q}$. Almost always, terms in $\mathcal{T}$ were tried in descending order; in some trials a number of terms from the top of the pool were preselected to form an initial current query; in a few trials the top N terms were randomized to test the effectiveness of the *RSV* ordering.

In general the selection procedures involved, for each term T in $\mathcal{T}$, trying the effect of adding T to $\mathcal{Q}$ if it was not in $\mathcal{Q}$; or removing T if it was already in $\mathcal{Q}$. Some of the trial runs used addition of terms only—once a term was in $\mathcal{Q}$ it remained for ever; but it was soon noticed that later removal was sometimes beneficial. To speed the procedures, terms were usually no longer tested after their addition or removal had failed to increase the score on more than some predetermined number (usually four) of successive iterations; in a few trials these "dropped" terms were re-examined occasionally, but this appears to be of only small benefit.

1 *Find best (FB).* In each iteration every undropped term was examined. At the end of the iteration the highest scoring term was added to or removed from $\mathcal{Q}$. Variants included *removing only* (when the initial query contained all the terms in $\mathcal{T}$); and *adding only*, when the initial query was empty or contained only a few terms.

2 *Choose first positive (CFP).* In each iteration the first term which increases the score by an amount greater than a predetermined threshold percentage is added to or removed from $\mathcal{Q}$. Obviously this goes more quickly than *FB*, but might be expected to give results which are less good. In fact, it appears to give slightly better results.

3 *Choose all positive (CAP).* Every term is considered and immediately added to or removed from the query if it increases the score by more than the threshold. An iteration ends when all non-dropped terms have been considered. Runtime is comparable with method *CFP* (fewer but much longer iterations).

A number of stopping criteria were built in, including elapsed time, maximum number of iterations and maximum query size, but in practice almost all runs ended at the point where there was no single term whose addition or removal increased the score.

4.4 Term selection databases

At least three databases may be used: one for the extraction of terms from relevant documents and their weighting and ranking, one for term selection and one for comparative evaluation of the queries. On the one hand, one needs to use as much of the available relevance information as possible, suggesting that the full database should be used for everything; but there is then a danger of over-fitting (a query consisting of a handful of rare terms from each of the known relevant documents might score very highly on its source database but poorly on any other). Probably there is a need for compromise, or the procedure used for an individual topic should depend on the amount of relevance information for that topic.

In choosing selection algorithms and scoring function for the official runs, we used the odd database both as the source for the term pool and for term selection, but the even database for comparison of methods. That is, the queries were entirely produced from the odd database but then evaluated on the even; the evaluation database thus being independent of the one used to derive the queries, although obviously very similar in makeup.

However, having chosen selection and scoring methods, the official queries were produced using the whole of the training database (as was done in TREC–3). Since the TREC–4 conference some inconclusive experiments have been done in an attempt to compare the effect of using various database combinations. Table 4 may suggest that, at least where there is a relatively large amount of relevance information, there is little difference between the four database combinations tried.

4.5 Routing experiments and discussion of results

In a general routing situation the object is, given a training database containing known relevant and nonrelevant documents, to find good methods of weighting and ordering the terms, finding suitable term pool sizes, scoring functions which are good predictors of results on the test database (which, it must be assumed, will be reasonably similar in makeup to the training database), and good algorithms for term selection from the pool. In our case, we disregarded the question of weighting and term ordering. The weighting follows the Robertson/Sparck Jones theory [3], and term ordering was always by Robertson Selection Value [4] (but term ordering affects some of the selection algorithms more than others).

This still leaves a very large number of combinations to explore and selection runs can be slow. A recent estimate gave a very rough figure of 500 cpu–days on a fastish machine for a rather minimal, even preliminary, exploration of the space. It follows that the pre-deadline decisions were based on hope and pants-seats as much as on hard evidence.[2]

Most of the trials reported in this section were done after the TREC–4 relevance judgments were available, using a subset (*T22*) of 22 of the topics[3] derived by selecting alternate topics and then eliminating three for which there were very few relevant documents in the test database. For the most part, the odd database (together with the topic statements and the "odd" relevant documents) was used to derive, weight and select terms; and the even database was used for evaluation. Evaluation was then checked using the test database. All evaluation runs were done at cutoff 1000 documents.

A t-statistic for the difference of mean score (i.e. average precision) between pairs of runs was calculated, the observations being the difference in score for each topic of the 22 sample topics. The validity of this procedure may be questioned, but it gives some indication of the likely significance of a result. It is noticeable that the between-topic variation is much higher than most between-treatment variation. The official results (Table 5) bear this out in the difference between the Cityr1 and Cityr2 runs: the former used the same procedures for all topics whereas for Cityr2 the best method was chosen for each topic. (There is also a large between-database component.) A full analysis of variance would need a very large amount of data to be useful.

Scoring functions

The initial object was to improve on the very inefficient and hence constrained procedure used in TREC–3, in which TREC average precision scores were used but with severe constraints on the number of terms considered and the number of successive "failures", no re-evaluation of a term after it has once failed to increase the score, and limits on runtime. The Brookes score was rapidly calculable, and it was hoped that it would correlate fairly well with some at least of the official TREC scoring measures. A few trials were done using the TREC–3 method to act as a baseline. It rapidly became clear that the Brookes score, at least when applied to a set with no cutoff, was a very poor predictor. It was less bad when evaluated on the top 1000 documents, or a small multiple of R, than on complete sets, but still gave poorer results than the TREC–3 procedure.

With hindsight it is obvious that at this stage we should have simply written a reasonably efficient procedure for calculating TREC average precision on the fly, but this was not done until after the TREC conference. Instead, a number of compromise measures were tried. In an attempt to make *BROOKES* more sensitive to the top end, the weights used in calculating the measure were raised to powers greater then one, but this made little difference. The un-normalized difference of means was tried: this gave results not significantly worse than *BROOKES*. The precision at R documents (*RPREC*) was tried, as it had been for TREC–3, but *RPREC* is not a good predictor of average precision; TREC–3 experience showed that average precision scoring resulted in higher values of *RPREC* than *RPREC* scoring. It also tended to get "stuck", with no single term's addition or removal giving an increase in score. Various types of precision averaging were then tried, taken over up to 20 points. Not surprisingly, these gave results which were nearer to TREC average precision, and one of them (granularity 20) was used for the official Cityr1 queries.

Finally, after the conference, real TREC average precisions were used, sometimes in linear combinations with *BROOKES* (*CBM1*). Some recent results are summarized in Table 2. On 1000 documents, *BROOKES* turns out not significantly inferior to *TREC–AVEP* when evaluated on the test database. However, it is greatly inferior on the training database (difference significant at 0.995), comparable with *RPREC*; it is not obvious how to account for this. Further trials are needed on larger term pools and more topics. The linear combination *CBM1* appears to be very similar to *TREC–AVEP*.

Selection algorithms

A summary comparison of selection algorithms is given in Table 3. It was expected that adding the "best" term at each iteration would give the best results, but this does not appear to be the case. There is surprisingly little difference between the methods, although when done on all the topics *CAP* is confirmed to be the best. It is something of a surprise that *FB* looks to be the worst; it is, unlike the other methods, independent of term order-

[2] Runtimes on an SS10 on 25 topics and a pool of 50 terms vary between about four and 15 hours depending on scoring and selection method; 50 topics and 200 terms can take up to a week. (Obtaining the term pools also takes many hours, but of course this only has to be done once.)

[3] 3, 14, 20, 28, 30, 32, 36, 41, 44, 46, 48, 66, 82, 94, 96, 103, 113, 117, 123, 142, 161, 174

Table 2: Automatic routing: comparison of term selection scoring functions

	Mean	Score		
Scoring function	query len	even db	test db	% increase
Baseline run: topic terms, no relevance information	30	0.279	0.288	0.0
topic terms from term pool	21	0.294	0.318	10.4
top 24 terms from term pool	24	0.266	0.343	19.1
top 50 terms	50	0.248	0.316	9.7
top 100 terms	100	0.221	0.286	-0.7
TREC–AVEP	24	0.378	0.402	39.6
AVEP, granularity 10	23	0.372	0.397	37.8
AVEP, granularity 3		0.359	0.389	35.1
BROOKES	22	0.347	0.394	36.8
DIFFM	23	0.340	0.379	31.6
3-PT	17	0.359	0.356	23.6
RPREC	12	0.342	0.340	18.1
CBM1	25	0.374	0.401	39.2

Topics: *T22.*
Terms from odd database; term pool 50 & selection algorithm *CAP* unless stated.
Selection on odd database and assessment on even and test databases.
Cutoff 1000 throughout. Scores are official TREC average precisions.

ing (within the term pool), and one would expect this to be beneficial. When observed—all these runs were logged in great detail—it seems to have something of a tendency to get stuck, easily reaching a stage where no single term gives any increase in score. The first and second *FB* rows in Table 3 are not significantly different from the first *CAP* row on the evaluation (even) database, but all three *FB* rows are significantly worse ($P = 0.9$) on the test database. *CFP* seems to lie between the other two methods. Retrying "failed" terms after a few iterations is occasionally beneficial. Disallowing term removal has a small but noticeable detrimental effect.

Official results

Table 5 gives some test database results on the whole topic set, including the official Cityr1 and Cityr2 runs. While City did fairly well again, there is still clearly a lot of scope for improvement. Both the runs are better (relative to other good runs) at the low-recall end than the high end. Both runs show average precision greater than or equal to the median of all the comparable runs on 44 out of 50 topics, but Cityr2 returned 49:1 on the median precision at 100 documents (and the "1" was topic 50, which had very few relevant documents).

Summary and conclusions

TREC–AVEP appears to be the best scoring function, perhaps not surprisingly as the evaluation was done on that measure. That aside, *TREC–AVEP* is quite a good predictor for other evaluation measures such as R-precision (e.g. one is likely to get higher mean R-precision by optimizing on average precision than on R-precision itself). The simple Brookes measure correlates only weakly with average precision, and gave poor evaluation results on the even database; hence it wasn't even considered for the official runs. However, when evaluated on the test database, results were not much inferior to *TREC–AVEP*, and a linear combination of the two seems to be about as good as the latter. This presumably has something to do with the large disparity between the two databases, and it seems doubtful whether *BROOKES* would be a good choice in a real-life routing situation.

As regards selection algorithm, *CAP* is probably the best, though not by very much. This is a surprising result which needs explaining. It suggests that the ordering of terms by *RSV* is of benefit. It doesn't do so well when the order of the terms in the pool is randomized. Varying the size of the initial set between zero and 50 terms didn't make much difference; sometimes, though, a non-empty initial set containing some of the "better" terms seems to render the selection procedure less likely to get prematurely "stuck". Increasing the number of terms in the pool was beneficial. Again, this may have been partly because it rendered the selection algorithm less likely to stick—it was noticeable that sometimes very low-ranking terms would be added at an early stage, only to be later removed.

On the question of databases, it has not been possible to reach even tentative conclusions. Retrospectively, the benefit of using all the relevance information appears to more than compensate for the dubious use of

Table 3: Automatic routing: effect of varying the selection method

Scoring	Mean	Score on		
method	query len	odd	even	test
CAP	24	0.449	0.378	0.402
CAP, add only	28	0.436	0.372	0.395
FB	20	0.438	0.370	0.390
FB with retry	20	0.441	0.371	0.390
FB, add only	19	0.428	0.370	0.385
CFP	25	0.444	0.373	0.392
CFP with retry	22	0.446	0.373	0.400
CFP, add only	27	0.429	0.370	0.393

Topics: *T22*.
Terms from odd database; term pool 50.
Selection on odd database. *TREC–AVEP* scoring.
Assessment on even and test databases.
Scores are official TREC average precisions.
Cutoff 1000 throughout.

Table 4: Automatic routing: effect of varying term source and selection databases

Terms	Selection	Mean	Score on			
from	on	query len	full	odd	even	test
full	full	24	0.397			0.402
full	odd	23		0.447	0.387	0.400
odd	full	23	0.389			0.396
odd	even	22	0.377		0.432	0.395

Topics: *T22*.
Term pool 50. *TREC–AVEP* scoring. *CAP* selection.
The scores are official TREC average precisions.
Cutoff 1000 throughout.

Table 5: Automatic routing: predictive results on test database, terms & selection on full training database

Scoring function	Selection algorithm	$\|\mathcal{T}\|$	Initial query	Notes	AveP	P5	P30	P100	R-Prec	Rcl
AVEP	varies	varies	varies	passages(4, 2, 32)	0.407	0.688	0.571	0.465	0.422	0.844
As row above, but no passages					0.390	0.688	0.573	0.442	0.406	0.832
AVEP	*CAP*	200	3	passages(1, 1, 20)	0.394	0.678	0.558	0.435	0.405	0.860
CBM1	*CAP*	200	3	no passages	0.390	0.668	0.583	0.443	0.413	0.825
AVEP	*CFP*	150	3	passages	0.389	0.700	0.559	0.440	0.405	0.834
AVEP	*CAP*	100	3	passages	0.387	0.684	0.569	0.444	0.408	0.829
As row above, but no passages					0.373	0.700	0.571	0.435	0.398	0.807
AVEP	*CAP*	150	3	no passages	0.378	0.720	0.577	0.434	0.401	0.811
AVEP	*CB*	100	3	no passages	0.319	0.640	0.530	0.385	0.356	0.751

The first row is the official Cityr2 and the sixth is Cityr1.

the same database both as term source and for term selection. Significantly the best test database results were obtained in this way. But using the same database is probably not a satisfactory way of obtaining evidence about the relative merits of the various scoring functions and selection procedures.

5 Non-interactive ad hoc

In TREC–3 City had some success with deriving the ad hoc queries from terms extracted from the top-ranked documents retrieved by a pilot search on the topic terms [1, Section 5]. It was not obvious that this technique, which must depend on reasonably high precision at the low end, would work with the very short TREC–4 ad hoc topics. It turned out, though, that such query expansion was still beneficial, leading to an increase of 20% or more in average precision (with perhaps a small decrease in mean low–end precision). Initial trials showed that results were still very poor compared with those obtained using the fuller topic statements, so it was decided to try the effect of some manual editing of the queries, both before and after expansion. The editing was done without any trial searches. The (unwritten) procedure was something like "Read the topic statement, then remove any term you think is likely to be detrimental in a search for documents about this topic; don't spend too long thinking about it".

Initial queries

Most of the trial runs were done using the only the DESCRIPTION fields of the TREC–4 routing topics on the disks 2 & 3 database (the live runs, of course, used the full [!] topics 202–250). In some runs, passage searching was used. For the manual runs, one of the team members pre-edited the topic statements by removing terms thought to be detrimental; no terms were added. For example, topic 207

> What are the prospects of the Quebec separatists achieving independence from the rest of Canada?

became

> Quebec separatists achieving independence Canada

In effect, "prospects" and "rest" were removed, because the other words would have been stopped in any case.

Query expansion

The top R documents from the pilot search were output and all terms other than stop and semi-stop terms extracted. These were $w^{(1)}$–weighted in accordance with their occurrence in the top documents, with topic terms loaded on the basis of their having occurred in r' out of R' fictional relevant documents (usually 19 out of 20, although more extreme loadings were also tried). The terms were then arranged by descending RSV value, and the required number taken from the top of the list, any term with RSV less than 3 being discarded.[4] For manually post-edited queries one of the team members then removed unwanted terms.

Results

A selection of trial results is in Table 6[5] and final results in Table 7.

It is clear that expansion was still beneficial despite the brief topics—compare the expansion runs in Table 6 with the first control row, in which no expansion was used. However, no expansion compensated for the absence of topic TITLE and CONCEPTS fields. Not surprisingly, the mean low precision is worse in the expansion runs, although it varies widely between topics. Trial results were rather flat over 20–100 documents and 15–30 terms. There is little or no evidence that the use of passages in the pilot search gave any improvement, nor that the topic–term loading was useful.

With regard to the manually edited queries, pre-editing gave a considerable improvement in the trial situation, but post-editing seems to have had a neutral effect in the trials and a detrimental one when used on topics 202–250. More investigation is needed, although it would be hard to obtain objective comparisons across different topic sets and editors.

6 Conclusions

The interactive track at TREC is beginning to bear fruit. What we need now is extensive diagnostic analyses, to fill out the evidence provided by the summary results.

The routing task continues to show that relatively heavy computation based on the training set can produce good results. More detailed conclusions about these methods are given in section 4.5 above.

References

[1] Robertson S E *et al.* Okapi at TREC–3. In: [6]. p109–126.

[4] In a few cases this resulted in the final query containing less than the specified number of terms.

[5] Note that the average precision, R–precision and recall figures in Table 6 are artificially low because we used the full set of disks 1, 2 & 3 relevance judgments instead of disks 2 & 3 only.

Table 6: Non-interactive ad hoc trials

Feedback docs	terms	Notes	AveP	P5	P30	P100	R-Prec	Rcl
Control runs								
0	0	no expansion, no passages	0.062	0.392	0.299	0.234	0.145	0.225
0	0	no expansion, no passages, topic fields TCND	0.119	0.492	0.414	0.347	0.221	0.349
50	20	manual pre-editing, final passages	0.100	0.396	0.350	0.295	0.194	0.309
50	≤ 20	manual pre- & post-editing, final passages	0.099	0.412	0.363	0.304	0.191	0.307
50	20	initial & final passages	0.092	0.312	0.289	0.277	0.187	0.301
50	20	final passages	0.089	0.356	0.321	0.273	0.178	0.286
50	20	no passages	0.087	0.352	0.318	0.274	0.176	0.282
50	20	no passages, no loading	0.086	0.384	0.316	0.269	0.175	0.280
50	15	final passages	0.090	0.332	0.316	0.275	0.182	0.287
30	20	no passages	0.086	0.336	0.313	0.272	0.176	0.280
30	20	no passages, no loading	0.086	0.348	0.311	0.272	0.175	0.281
50	30	no passages, no loading	0.086	0.364	0.314	0.270	0.173	0.281
50	10	no passages	0.085	0.336	0.312	0.266	0.175	0.269
100	20	no passages, no loading	0.081	0.324	0.283	0.255	0.166	0.270
100	30	no passages, no loading	0.079	0.320	0.280	0.255	0.163	0.264
30	50	no passages	0.080	0.300	0.294	0.261	0.171	0.274
DB disks 2 & 3, TREC–4 routing topics, topic DESC field only & 19/20 loading except where stated								

Table 7: Non-interactive ad hoc results

Feedback docs	terms	Notes	AveP	P5	P30	P100	R-Prec	Rcl
50	20	initial & final passages	0.257	0.522	0.394	0.274	0.298	0.586
50	20	initial passages	0.250	0.535	0.389	0.271	0.298	0.581
50	≤ 30	manual pre- & post-edit, final passages	0.226	0.498	0.368	0.254	0.271	0.544
0	0	final passages, no expansion	0.209	0.502	0.342	0.235	0.265	0.531
0	0	no passages, no expansion, manual pre-edit	0.215	0.518	0.350	0.242	0.279	0.536
0	0	no passages, no expansion	0.207	0.526	0.349	0.238	0.276	0.529
The top row in the official Cityal and the third row Cityml DB disks 2 & 3, topics 202–250, 19/20 topic term loading								

[2] Brookes B C. The measure of information retrieval effectiveness proposed by Swets. *Journal of Documentation 24* 1968 p41–54.

[3] Robertson S E and Sparck Jones K. Relevance weighting of search terms. *Journal of the American Society for Information Science 27* May–June 1976 p129–146.

[4] Robertson S E. On term selection for query expansion. *Journal of Documentation 46* Dec 1990 p359–364.

[5] Robertson S E *et al.* Okapi at TREC–2. In: [7]. p21–34.

[6] *Overview of the Third Text REtrieval Conference (TREC–3)*. Edited by D K Harman. Gaithersburg, MD: NIST, April 1995.

[7] *The Second Text REtrieval Conference (TREC–2)*. Edited by D K Harman. Gaithersburg, MD: NIST, 1994.

A Interactive system description

A.1 GUI interface

The interface was a GUI to the BSS written in C, C++ and TCL/TK. Figures 2 and 3 show screen dumps of the running system.

The Interface was composed of five main parts.

A.1.1 A query entry box

A.1.2 A query terms display box

A scrollable listbox in which were displayed the ranked list of current query terms.

A.1.3 A hitlist display box

A scrollable listbox in which were displayed the ranked hitlist for a given search.

A.1.4 A relevance judgments display box

A scrollable listbox in which were displayed the ranked list of documents judged as relevant by the user.

Items A.1.1 to A.1.4 were all displayed in one window together with three context-sensitive buttons to:

- SEARCH

  Search on the current query iteration—only selectable once a query had been defined.

- EXPAND

  Expand the current query iteration—only selectable once positive relevance judgments had been made.

- END SEARCH

  Selectable at any stage in the search.

A.1.5 A full record display box

A pop-up text window in which full records were displayed. At the bottom of this window were three buttons for making relevance judgments. These are described in more detail in Section A.2.4.

A.2 User interaction

A.2.1 User input of query terms

Users entered one or more terms, optionally followed by an adjacency operator as the last non-space character in the line, into the query entry box.

- No adjacency operator. Each term was treated as a single member of the query.

- An adjacency operator. The set of terms were treated, in input order, as both a phrase (ADJ), possibly including intervening stopwords, and as 'within the same sentence' (SAMES). For example, the input line "laws of the sea" would find documents that included, among others, both "laws of the sea" and "Soviet law does not provide for the right of innocent passage in Black Sea waters". Any set of terms input with an adjacency operator will be referred to as a 'phrase'. Internally two sets were generated, S(A) and S(S), with number of postings and weights N(A), W(A) and N(S), W(S) respectively. These were then combined into one "unit" according to the rules:

Sets generated	Use
1. N(A) = N(S) = 0	discard both
2. N(A) = 0, N(S) > 0	S(S), N(S), W(S)
3. N(A) = N(S), N(A) > 0	S(A), N(A), W(A)
4. N(A) > 0 and N(A) < N(S)	S(A), N(A), W(A) and S(S)-S(A), N(S)-N(A), W(S). Count as one term only.

The weight calculated for each term—user terms, user 'phrases', and system generated terms—was a Robertson/Sparck Jones F4 predictive weight, with halves. In addition, user entered terms were given a loading; little_r and big_R increased by 4 and 5 respectively.

Terms were displayed in the query listbox in descending order of Robertson Selection value.

A.2.2 Searching

The top N terms from the current query (N <= 20) were combined in a best match operation (bm25). Passage retrieval (Section 2.1) was applied to the document set generated with parameters p_unit = 4, p_step = 2, k1 = 1.6 and b = 0.7. The weight of the document was taken to be the higher of the weight of the full record or the best passage.

A.2.3 Hitlist generation

For each search, the hitlist of documents to "view" was made up of the top 50 ranked unseen documents, i.e. not including any that had been judged from any previous document set. An entry for each document consisted of:

- A header line.
  ⟨record_no⟩ ⟨docid⟩ ⟨weight⟩ ⟨document_length⟩ [⟨passage_length⟩] ⟨passage_length⟩
  was shown only for documents longer than 10K characters.

- A system generated title.
  Since, except for ZF and PT documents, records

had no distinct title field, a "title" for display in the hitlist window was made up from approximately the first 150 characters from the headline field (if any) and the text field.

- Query term occurrence information

A.2.4 "Seeing" documents

Documents were selected for "seeing" by double-clicking on the appropriate header line in the hitlist window. Each document was displayed in a pop-up, scrollable text window. The best passage, or the whole document if the same, was highlighted in cyan; query terms were highlighted in green. The document was displayed either at the start line of the best passage, or the line containing the first query term if there was no best passage.

In the case of documents longer than 10K characters, only the best passage was displayed with two or three lines of context around the start and end of the passage.

At the bottom of the text window were three buttons—"YES", "PASSAGE" and "NO" —to allow users to make a relevance judgment.

- "YES"

  Relevant: terms were extracted from the text field of the entire document.

- "PASSAGE"

  Relevant: terms were extracted only from the best passage.

- "NO"

  Not relevant.

Searchers had to make a relevance judgment before they could go onto to any other part of the interface.

A.2.5 Relevance judgments pool

The ranked hitlist information for all documents currently judged as relevant. Any member of the current relevance judgments pool that existed in a document set generated by a new search, had its weight set to that which it had in the latest document set; the display order was adjusted accordingly.

A.2.6 Expanding a query

Once the searcher had made one or more positive relevance judgments—"YES" or "PASSAGE" —the EXPAND button was selectable. Clicking on EXPAND caused the current query to be merged with all terms extracted from new relevant documents with relevance information adjusted accordingly and new weights and *RSV*s calculated. These terms were displayed in the query listbox in descending order of Robertson Selection value with a line drawn under the 20th term.

A.2.7 Removing terms

Terms were removed from the current query by double clicking on its entry in the query listbox. Although any term could be removed from the current query, its main use was for the removal of noise generated during the extraction of terms from relevant documents.

A.2.8 Quitting

Quitting was a two stage process performed by clicking the "END SEARCH" button twice.

- END SEARCH (1).

  This was taken as the end of the allowable search time, i.e. the end of the primary task. The top 20 terms from the current query iteration were displayed in the query listbox. Searchers were then allowed to modify this query by removing terms and / or adding new terms in the query entry box.

- END SEARCH (2).

  This marked the start of the secondary task, i.e. the generation of the ranked list of 1000 documents.

B Experimental Conditions

B.1 Searcher characteristics

Five (female) searchers were randomly assigned five searches each which were undertaken over a two day period. Three were research staff on the Okapi team and two were research students. Their ages ranged from late 20s to early 50s. All had extensive experience with the system but the interface was designed specifically for the interactive track and three of the searchers contributed to its specification.

None of the searchers had any specialist knowledge of any of the subject domains covered by the topics, they played the role of an intermediary searching on behalf of a remote end-user.

B.2 Task description/training

All but one of the searchers had participated in the TREC–3 interactive routing task, and all were aware of the different nature of the task definition for TREC–4. The searchers were given the opportunity to familiarize themselves with the interface by carrying out three or four trial searchers on the 25 non-interactive TREC–4 topics, over a period of a week prior to the actual test sessions.

Before conducting the test searches, the participants were issued with a set of guidelines which included information such as: the task definition as defined for the interactive track, procedures to follow in the event of a system crash, as well as suggestions on strategies which could be adopted for carrying out searches, e.g. to remove terms deemed as irrelevant from extracted lists of terms before carrying out an expanded search.

Searchers were also asked to fill in a search evaluation questionnaire for each search before proceeding on to the next search.

C Search process

C.1 Clock time

Times are given to the nearest tenth of a minute.

Mean	Median	Variance	Range[6]
30.8	30.6	12.3	22.0–41.7
30.3	30.6	7.5	22.0–34.3

C.2 Number of documents viewed (hitlist) and seen (full text)

C.2.1 Number of documents viewed

"Viewing" a document consisted of seeing a header line, a system generated title, and query term occurrence information as described in Appendix A.2.3.

The figures represent the percentage distance scrolled through the hitlist by the searcher.

Mean	Median	Variance	Range
54.18	20	1173.34	2–100

C.2.2 Number of documents seen.

"Seeing" a document consisted of showing the full record in a scrollable window.

Mean	Median	Variance	Range
31.12	30	97.2	13–52

C.3 Number of iterations

An iteration, i.e. a new query formulation, was taken to be marked by each 'search' command.

Mean	Median	Variance	Range
3.56	3	2.01	1–7

No. expands	Queries
0	3
1	20
2	2

C.4 Number of terms used in queries

In all queries N = no adjacency and A = Adjacency.

C.4.1 Initial interactive

Type	Mean	Median	Variance	Range
N	2.32	1	4.98	0–7
A	2.40	2	3.83	0–8
All	4.72	4	8.96	1–12

C.4.2 Final interactive (primary task)

Type	Mean	Median	Variance	Range
N	12.84	13	33.72	1–20
A	3.80	3	7.08	0–10
All	16.64	19	25.74	3–20

C.4.3 Offline query (secondary task)

Type	Mean	Median	Variance	Range
N	12.80	13	33.83	0–20
A	4.24	3	8.02	0–10
All	17.04	19	26.87	3–20

C.4.4 "Phrases" defined by searchers

Phrases generated:	133
Phrases used:	121

C.5 Use of system features

+ terms defined with an adjacency operator

N terms defined with no adjacency operator

A all terms defined

Command	Mean	Median	Variance	Range
define - +	6.04	7	15.46	0–15
N	3.88	3	10.44	0–12
A	9.92	10	19.74	3–22
search	3.56	3	2.01	1–7
show	31.12	30	97.19	13–52
expand	0.96	1	0.21	0–2
remove	39.32	26	1146.14	0–113

C.6 Number of user errors

Data on user errors were not collected. However, there were three searches during which the system "crashed". These were undertaken by another searcher.

C.7 Search narrative for topic 236

The initial query consisted of the words 'sea', 'laws', 'disagree' and the adjacency term 'naval laws'. None of the documents resulting from this search were examined by the user since she could see from the hitlist that

[6] The maximum value of 41.7 was almost completely due to having to wait for around 30 minutes to view an FR record. The second row of figures have been calculated excluding this extreme value.

they were not relevant to the query. The user says of her lack of success: "In retrospect this was because I didn't know what I was looking for . . . I could interpret the question—that it wanted information on disagreements on maritime law—but I wasn't sure what the disagreements might be—about age of ships, uniforms etc."

Two query terms were removed: 'disagree' and 'sea' and the user searched once again.

This time the user examined the first document and rejected it as not being relevant to the query. She then removed the term 'law' and searched again.

From the second document set five documents were looked at and one selected as relevant. Following this, the term 'naval laws' was removed from the query, leaving the query as the phrase 'laws of the sea'. The user searched again and selected two out of the five document seen as relevant. Following this she expanded her search. She gave as a reason for doing this: "I had found several relevant documents and had no idea what new terms to add myself so I thought it would be useful."

Of the expanded terms, the user deleted 46 terms. These included mainly proper names—surnames and the names of countries: 'Leslye', 'Arsht', 'Mariana', 'Samoa' and so on. Another search was then done and of 24 documents seen, six whole documents were selected as relevant, Seven relevant passages were selected and 11 documents were rejected. The user then expanded her search again.

Of the terms brought up by the second expansion, 60 were rejected. As with the terms rejected following the first expansion, these were mainly proper names, including: 'Gerasimov', 'Sevastopol', 'Kamchatka', 'McNaught'. The results of this search show the largest number of rejected terms (113) of any search session in the round. This may be due to the nature of the topic: most documents found concerned individual incidents of disagreement between two countries about sea territory and for this reason there were many geographical names in each document.

The user then ended the session and altered the final term-set first by deleting the term 'claims' and then by adding the term 'violated'. This left the following twenty terms as the final term-set:

Waters, laws of the sea, territory, miles, coast, seas, passage, warships, admiralty, seamounts, vessels, shipwreck, unassigned, eastward, Navy, offshore, maneuvred, ships, maritime, violated

The search took just under 26 minutes. Altogether, 35 documents were examined, of which nine whole documents and seven passages were judged as relevant. Most relevant documents referred to incidents where one country had violated the laws of the sea by crossing into another country's territory.

This search was one which started off badly with the user not sure what to look for but which picked up when the user expanded with just a couple of relevant documents and could see the type of information that the query was referring to.

Following is a breakdown of command usage on this search.

- Define(N): single term(s)
- Define(A): a "phrase"

Define(N)	Define(A)	Search	Show	Docset	Remove	Expand
3	2	6	35	6	113	2

D Search Evaluation: questionnaire results

1. How difficult was it to interpret the topic?

Easy	13	52%
Moderately easy	9	36%
Difficult	3	12%
Very difficult	0	0%
Total	25	100%

2. How difficult was it to generate initial search terms?

Easy	7	28%
Moderately easy	11	44%
Difficult	6	24%
Very difficult	1	4%
Total	25	100%

3. How difficult was it to find relevant items from the initial hitlist?

Easy	6	24%
Moderately easy	7	28%
Difficult	7	28%
Very difficult	5	20%
Total	25	100%

4. If you added new search terms in the course of the search, explain why?

Did not add terms	7	28%
Added terms to improve recall	14	56%
Added terms to improve precision	4	16%
Total	25	100%

5. If you chose to expand the search what led you to do so?

Did not expand	3	12%
Expanded to find more of the same type of item	11	44%
Expanded to find more precise items	5	20%
Expanded out of desperation	6	24%
Total	25	100%

6. If you removed terms from the extracted term lists on what basis did you do so?

Proper names	16	47%
Numbers	8	24%
Difficult	8	24%
Very difficult	2	5%

7. Was it difficult to determine what would constitute the offline query?

Easy	13	52%
Moderately easy	8	32%
Difficult	2	8%
Very difficult	2	8%
Total	25	100%

8. How would you rate the overall difficulty of the topic question?

Easy	8	32%
Moderately easy	7	28%
Difficult	7	28%
Very difficult	3	12%
Total	25	100%

9. How would you rate the success of your search?

Successful	9	36%
Moderately successful	9	36%
Not successful	7	28%
Total	25	100%

Table 8: Primary task evaluation results

Topic	Relevant	Retrieved	Rel_ret	Time	Precision	Recall	Rel/min
202	283	36	24	30	0.6667	0.0848	0.8000
203	33	8	4	30	0.5000	0.1212	0.1333
204	397	21	17	34	0.8095	0.0428	0.5000
205	310	19	9	29	0.4737	0.0290	0.3103
206	47	4	2	32	0.5000	0.0426	0.0625
207	74	14	11	31	0.7857	0.1486	0.3548
208	54	3	1	33	0.3333	0.0185	0.0303
209	87	7	1	32	0.1429	0.0115	0.0313
210	57	36	31	30	0.8611	0.5439	1.0333
211	323	19	19	31	1.0000	0.0588	0.6129
212	153	24	24	30	1.0000	0.1569	0.8000
213	21	20	10	31	0.5000	0.4762	0.3226
214	5	3	3	28	1.0000	0.6000	0.1071
215	183	13	10	31	0.7692	0.0546	0.3226
216	36	28	16	31	0.5714	0.4444	0.5161
220	24	1	1	21	1.0000	0.0417	0.0476
223	363	15	10	29	0.6667	0.0275	0.3448
227	347	42	36	30	0.8571	0.1037	1.2000
232	9	2	0	27	0.0000	0.0000	0.0000
236	43	16	11	24	0.6875	0.2558	0.4583
238	270	12	9	41	0.7500	0.0333	0.2195
239	123	16	13	30	0.8125	0.1057	0.4333
242	38	9	9	30	1.0000	0.2368	0.3000
243	69	41	4	30	0.0976	0.0580	0.1333
250	86	16	11	33	0.6875	0.1279	0.3333
All	137.40	17.00	11.44	30.3	0.6589	0.1530	0.3763
Micro average					0.6729	0.0833	

Table 9: Time boundaries

Measures calculated at 5 mins, 10 mins, 20 mins and at the end of the search.
Average end time = 31 mins.

D: time in minutes

J: number of iterations

E: number of official relevance judgments

F: number of searcher relevance judgments

G: number of official rels chosen by the searcher

H: precision (G / F)

I: recall (G / E)

K: number of full records seen by the searcher

L: number of official rels seen by the searcher as full records

M: number of official rels seen and rejected by the searcher

N: proportion of seen official rels that were rejected (M / L)

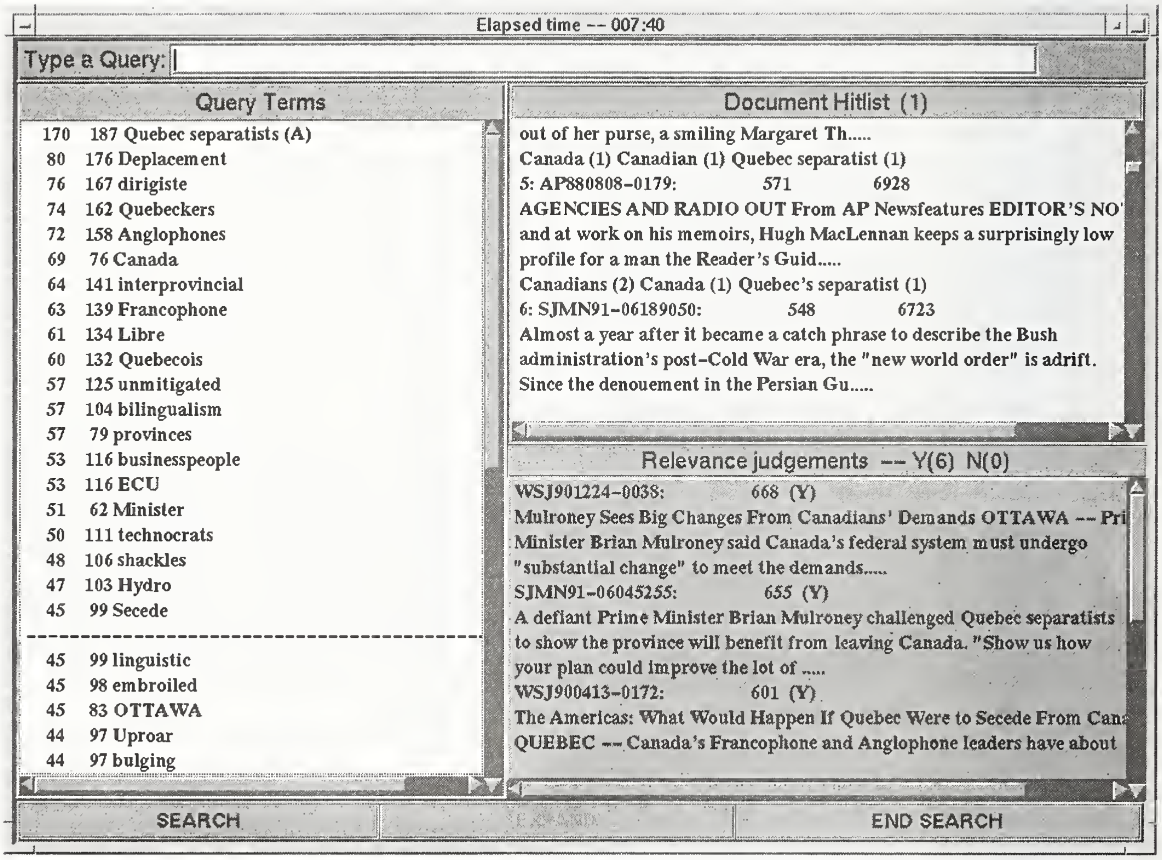
O: proportion of seen documents chosen by the searcher (F / K)

D	J	E	F	G	H	I	K	L	M	N	O
5	1.32	137.4	1.04	0.60	0.2600	0.0099	2.72	1.04	0.44	0.2500	0.1700
10	1.68	137.4	3.32	2.32	0.6260	0.0510	8.12	3.72	1.40	0.2742	0.2693
20	2.60	137.4	9.88	7.32	0.6807	0.1082	20.12	9.48	2.16	0.2400	0.3482
End	3.56	137.4	17.00	11.44	0.6589	0.1530	31.12	14.52	3.08	0.2447	0.3518
Micro average					0.6729	0.0833				0.2121	0.5463

Table 10: Iteration boundaries

Measures calculated at each iteration. The maximum number of iterations during any search was 7. For searches that had n iterations, n < 7, the values at iteration n were used for iterations n + 1 to 7.

J	D	E	F	G	H	I	K	L	M	N	O
1	2.64	137.4	0.00	0.00	0.0000	0.0000	0.00	0.00	0.00	0.0000	0.0000
2	14.32	137.4	5.88	4.20	0.5083	0.0355	11.44	5.52	1.32	0.2219	0.2786
3	20.92	137.4	10.16	6.56	0.6022	0.1022	18.88	8.72	2.16	0.2339	0.3065
4	25.24	137.4	13.04	8.84	0.6483	0.1198	23.80	11.44	2.60	0.2858	0.3377
5	28.20	137.4	15.76	10.72	0.6461	0.1395	27.96	13.72	3.00	0.2799	0.3400
6	29.44	137.4	16.56	11.08	0.6537	0.1446	29.56	14.12	3.04	0.2518	0.3465
7	30.24	137.4	17.00	11.44	0.6589	0.1530	30.80	14.52	3.08	0.2447	0.3518
Micro average					0.6729	0.0833				0.2121	0.5519

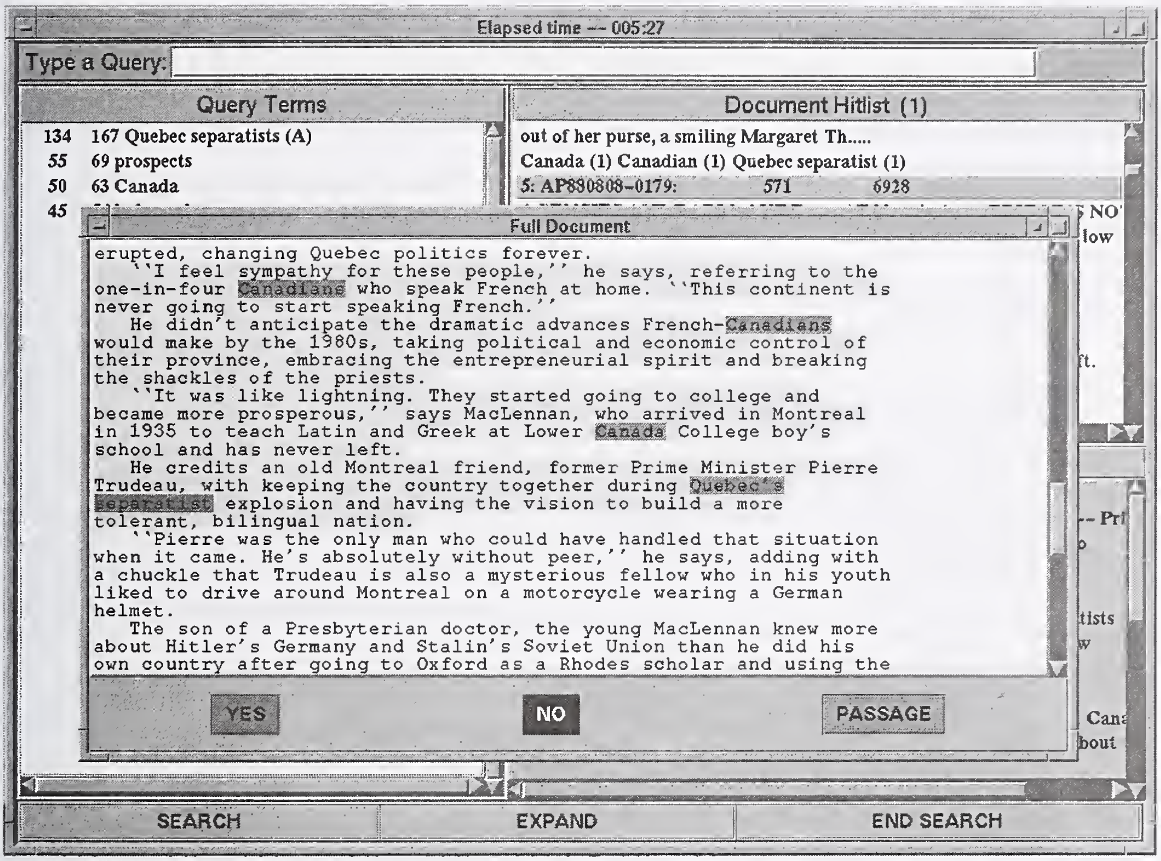


```
170   187 Quebec separatists (A)
 80   176 Deplacement
 76   167 dirigiste
 74   162 Quebeckers
 72   158 Anglophones
 69    76 Canada
 64   141 interprovincial
 63   139 Francophone
 61   134 Libre
 60   132 Quebecois
 57   125 unmitigated
 57   104 bilingualism
 57    79 provinces
 53   116 businesspeople
 53   116 ECU
 51    62 Minister
 50   111 technocrats
 48   106 shackles
 47   103 Hydro
 45    99 Secede
------------------------------------------
 45    99 linguistic
 45    98 embroiled
 45    83 OTTAWA
 44    97 Uproar
 44    97 bulging
```

```
out of her purse, a smiling Margaret Th.....
Canada (1) Canadian (1) Quebec separatist (1)
5: AP880808-0179:            571            6928
AGENCIES AND RADIO OUT From AP Newsfeatures EDITOR'S NO
and at work on his memoirs, Hugh MacLennan keeps a surprisingly low
profile for a man the Reader's Guid.....
Canadians (2) Canada (1) Quebec's separatist (1)
6: SJMN91-06189050:          548            6723
Almost a year after it became a catch phrase to describe the Bush
administration's post-Cold War era, the "new world order" is adrift.
Since the denouement in the Persian Gu.....
```

```
WSJ901224-0038:          668 (Y)
Mulroney Sees Big Changes From Canadians' Demands OTTAWA -- Pri
Minister Brian Mulroney said Canada's federal system must undergo
"substantial change" to meet the demands.....
SJMN91-06045255:         655 (Y)
A defiant Prime Minister Brian Mulroney challenged Quebec separatists
to show the province will benefit from leaving Canada. "Show us how
your plan could improve the lot of .....
WSJ900413-0172:          601 (Y)
The Americas: What Would Happen If Quebec Were to Secede From Cana
QUEBEC -- Canada's Francophone and Anglophone leaders have about
```

Figure 2: Interactive interface: main screen



134	167 Quebec separatists (A)
55	69 prospects
50	63 Canada
45	

out of her purse, a smiling Margaret Th.....
Canada (1) Canadian (1) Quebec separatist (1)
5: AP880808-0179: 571 6928

```
erupted, changing Quebec politics forever.
   ``I feel sympathy for these people,'' he says, referring to the
one-in-four Canadians who speak French at home. ``This continent is
never going to start speaking French.''
   He didn't anticipate the dramatic advances French-Canadians
would make by the 1980s, taking political and economic control of
their province, embracing the entrepreneurial spirit and breaking
the shackles of the priests.
   ``It was like lightning. They started going to college and
became more prosperous,'' says MacLennan, who arrived in Montreal
in 1935 to teach Latin and Greek at Lower Canada College boy's
school and has never left.
   He credits an old Montreal friend, former Prime Minister Pierre
Trudeau, with keeping the country together during Quebec's
separatist explosion and having the vision to build a more
tolerant, bilingual nation.
   ``Pierre was the only man who could have handled that situation
when it came. He's absolutely without peer,'' he says, adding with
a chuckle that Trudeau is also a mysterious fellow who in his youth
liked to drive around Montreal on a motorcycle wearing a German
helmet.
   The son of a Presbyterian doctor, the young MacLennan knew more
about Hitler's Germany and Stalin's Soviet Union than he did his
own country after going to Oxford as a Rhodes scholar and using the
```

Figure 3: Interactive interface: full record display

E Guidelines for Searchers

Okapi TREC Interactive Searching Guidelines

Each searcher will be assigned five searches to be undertaken in the given order with a sufficient break in between each. The search topic must not be viewed before the search session itself. The time allowed is around 30 minutes for the primary task. The lapsed time during a session is displayed on the screen.

In the event of a system crash when a search is in progress, the search must be passed on to another searcher.

The primary task or objective is to find as many relevant documents during the search session for the given topic without too much rubbish and using whatever strategy is available in Okapi and which you consider to be appropriate.

Some suggestions on how to proceed follow.

- Since the topics consist of short questions it is suggested to generate as many terms as possible for the initial search.
- It may be useful to indicate phrases wherever possible by typing a '+' after consecutive terms.
- Once a hitlist has been generated and items viewed, it may be fruitful to add further terms or phrases at this stage before any query expansion is undertaken. Trying to add your own new terms to a list of extracted terms once query expansion has been chosen, may not be as productive since their inclusion will depend on their relative weights.
- It is usually more effective to expand a search only after several items have been deemed relevant. If you want to expand a search, you must allow enough time, e.g. before the final ten minutes.
- The prime purpose of the relevance judgment is to indicate the relevance of an item to the topic or question. Making relevance judgments for the purpose of query expansion alone is not encouraged.
- Remove any terms deemed irrelevant from extracted lists of terms before carrying out an expanded search. The top twenty terms will be used to generate a new hit list.
- To determine the final query for the secondary task, it may be appropriate to select the final list of terms, your original query only or a combination of both including any new terms you may want to add at that point.

Please fill in a search evaluation on completion of a search before proceeding to the next one.

Xerox Site Report: Four TREC-4 Tracks *

Marti Hearst, Jan Pedersen, Peter Pirolli and **Hinrich Schütze**
Xerox Palo Alto Research Center
3333 Coyote Hill Road, Palo Alto, CA 94304
{hearst,pedersen,pirolli,schuetze}@parc.xerox.com

Gregory Grefenstette and **David Hull**
Rank Xerox Research Centre
6, chemin de Maupertuis, 38240 Meylan, France
{gregory.grefenstette,david.hull}@xerox.fr

1 Overview

The Xerox research centers participated in four TREC-4 activities: the routing task, the filtering track, the Spanish track, and the interactive track. We addressed the core routing task as a problem in statistical classification: given a training set of judged documents, build an error-minimizing statistical classifier to assess the relevance of new test documents. This year, we built on the methodology developed in [21] by adding a combination strategy that pooled evidence across a number of separately trained classification schemes. Since many of our classifiers infer probability of relevance, adapting our routing methods to the filtering track consisted of obtaining probability estimates for the remaining classifiers and reporting those documents scoring above the probability thresholds determined by the three set linear utility functions.

Our contribution to the Spanish track focussed on the effect of principled language analysis on a baseline retrieval system. We employed finite-state morphology [14] and hidden-Markov-model-based part-of-speech tagging [7] to analyze Spanish language text into canonical stemmed forms, and to identify verbs and noun phrases. Various combinations of these were then fed into SMART [1] for ranked retrieval.

This year our activity on the ad hoc task focussed on the interactive track, which allows arbitrary user interaction in the process of finding relevant documents. We developed a graphical user interface to two interactive tools, Scatter/Gather [6] and Tilebars [11], and asked a number of subjects to use this tool to "find as many good documents as you can for a topic, in around 30 minutes, without collecting too much rubbish." We set up an experimental design to measure the value of each tool, and their combination, averaging out subject effects. That is, we were interested in determining how well the average user might perform with interactive tools rather than measuring the very best performance possible assuming an expert searcher.

These efforts are described in more detail in the following sections.

2 The Routing Problem

The routing task can be treated as a problem of machine learning or statistical classification. A classification tool is inferred from the training set of judged documents and is used to predict the relevance of newly arriving documents. Traditional learning algorithms must be adapted due to the large scale of this problem. For example, one cannot use the full collection vocabulary as a feature set without overfitting to the training documents. Similarly, the full training set is simply too large, and relevant documents too rare to efficiently learn the optimal classification rule. Therefore, Xerox has developed a special three-step algorithm to solve the routing problem (described in detail in [21]). In this section, we summarize the Xerox routing strategy and present our new work on method combination developed for TREC-4.

2.1 Step 1: Local Regions

The document collection is parsed, tokenized, stemmed, and stop words are removed using the Text Database System (TDB) developed at Xerox PARC [8]. Indexed *terms* consist of single words and two-word phrases that occur over five times in the corpus (where a phrase is defined as an adjacent word pair, not including stop words). This process produces over 2.5 million *terms*. Then, each document is partitioned into overlapping sections of average length 300 words with an average overlap of approximately 250 words.

*Authors listed in alphabetic order. Hull, Pedersen and Schütze worked on the routing task, Hull on the filtering track, Grefenstette and Hull on the Spanish track, and Hearst, Pedersen, and Pirolli on the interactive track.

In the first stage, expanded queries are constructed using a modified version of the Rocchio technique for relevance feedback [3]. The expanded query is defined as the vector sum of the relevant documents in the training set. The original query is also included and given a weight of five relevant documents. No negative feedback is used, i.e. non-relevant documents are ignored. All documents in the collection are ranked according to their similarity to the expanded query, and the top 2000 documents are selected for each query. These documents define the *local region* for that query. Documents are given the rank of their highest scoring section and for all future analysis each document in the local region is represented only by this section.

There are a number of advantages to using only the local region in subsequent stages of the routing task. First, the size of the training set is substantially reduced, so it is possible to solve the problem using computationally intensive variable selection techniques and learning algorithms in a reasonable length of time. Second, the density of relevant documents is much higher in the local region than in the training collection as a whole, which should improve classifier performance. Third, the non-relevant documents selected for training are those which are most difficult to distinguish from the relevant documents. These non-relevant documents are clearly among the most valuable ones to use as training data for a learning algorithm.

2.2 Step 2: Document Representations

In the vector space model, one dimension is reserved for each unique term in the collection. Standard classification techniques cannot operate in such a high dimensional space, due to insufficient training data and computational restrictions. Therefore, some form of dimensionality reduction must be considered, even after applying the preliminary filtering step to construct the local region. We use two distinctive approaches to dimensionality reduction, optimal term selection and reparameterization of the document space.

The process of optimal term selection consists of identifying the words that are most closely associated with relevance. Our approach is to apply a χ^2 test to the contingency table containing the number of relevant and non-relevant documents in which the term occurs (N_{r+} and N_{n+}, respectively), and the number of relevant and non-relevant documents in which the term doesn't occur (N_{r-} and N_{n-}, respectively).

$$\chi^2 = \frac{N(N_{r+}N_{n-} - N_{r-}N_{n+})^2}{(N_{r+} + N_{r-})(N_{n+} + N_{n-})(N_{r+} + N_{n+})(N_{r-} + N_{n-})}$$

The higher the score, the greater the association between that term and the relevant documents. We select the 200 terms with the highest scores for each query to use with our learning algorithms.

As an alternative approach, we apply Latent Semantic Indexing (LSI) [9] to represent documents by a low-dimensional linear combination of orthogonal indexing variables. The LSI solution is computed separately for each query by applying a singular value decomposition to the sparse document by term matrix constructed from the local region. We then select the 100 most important LSI factors to serve as an alternative document representation. Optimal term selection works best when there is a relatively small specific vocabulary associated with the topic while LSI performs well when the topic has a very large vocabulary which can be organized into a smaller number of general concepts.

2.3 Step 3: Classification Algorithms

We have examined a number of classification techniques for the routing problem, including logistic regression, nearest neighbors, linear discriminant analysis (LDA), and neural networks. In the past, we have achieved the best performance with LDA (using LSI features) and a linear neural network (using both LSI and term features). For TREC-4, we decided to try to improve performance by combining the results of a number of different classification techniques. We submitted two runs, a baseline using LDA and a combination run using using Rocchio expansion, nearest neighbors, LDA, and a linear neural network fitting a logistic model.

We briefly describe the three classification techniques. Our variant of Rocchio expansion is detailed in step 1 above. Linear Discriminant Analysis finds a linear combination a of the feature elements which maximizes the separation between the centroids of the relevant $\bar{x}_r$ and non-relevant $\bar{x}_n$ documents: $a = S^{-1}(\bar{x}_r - \bar{x}_n)$, where S is the pooled within-group covariance matrix. The linear neural network (no hidden units) uses backpropagation to iteratively fit a logistic model. The network is not iterated to convergence, rather a validation set is used to determine the optimal number of training iterations, as a protection against overfitting. Our nearest neighbor technique is a slightly modified version of Yang's Expert Network [22]. Essentially, each new document is assigned a score equal to the sum of its similarity scores with respect to all relevant documents among its 50 nearest neighbors in the training set.

In order to combine results, the output of the various classification techniques needs to be normalized to the same scale. We accomplish this by converting the output for each classifier into an estimated probability of relevance. The probability of relevance of each document can be extracted automatically from the model for LDA and the network fitting a logistic model. For the nearest neighbor technique and Rocchio expansion, we used logistic regression to transform the output values into probability estimates [16], i.e. $p(s) = \frac{exp(a+bs)}{1+exp(a+bs)}$ where a and

b are the parameter estimates obtained from the training data.

We examined three possible approaches to combining evidence.

(1) Averaging the probability estimates.

(2) Using linear regression on the training data to determine the optimal linear combination of the probability estimates on a per-query basis.

(3) As (2), but compute the average optimal combination and use it for all queries.

In preliminary experiments, the simple average of the probability estimates (1) worked best, and we used this approach in our submitted run.

2.4 Summary of Routing Algorithm

We present a simple flow chart that describes the training and testing process of our routing algorithm. The training process:

- Compute 2000 nearest documents to Rocchio expanded query (local region).
- Segment documents and select segment most similar to expanded query (on a per-query basis).
- Compute LSI decomposition of local region and select 100 largest factors.
- Compute χ^2 statistic to select 200 most valuable terms in local region.
- Apply classification techniques to documents in local region of training set using the appropriate representation.
- Convert the document scores to probability estimates.
- Average the probability estimates for the combination run.

The testing process:

- Select test documents whose best segment exceeds a threshold similarity score.
- Obtain LSI document vectors for selected test documents from LSI term representation.
- Score selected test documents using the classifiers and convert the scores to probability estimates.
- Average the probability estimates for the combination run.
- Rank the test documents by descending probability score. All documents in the selected region are ranked ahead of those that fall below the threshold similarity score.

2.5 TREC-4 Results

The basic Xerox TREC-4 routing results are presented in Table 1. The first two rows measure uninterpolated average precision at all relevant documents (all) while the second pair measure average precision at 20 documents retrieved (20). The column *thr* gives the threshold used to select test documents for classification. For example, a threshold of 750 means that all documents which have a similarity score greater than the 750th training document (wrt the Rocchio expanded query) are selected. The other columns are: Roc = Rocchio expansion, NN = nearest neighbors, LDA = linear discriminant analysis, Net = logistic neural network, and Cmb = combination run. The submitted runs are marked in the table. The Cmb run submitted to NIST was partially corrupted due to a programming error. The corrected results are presented in the table.

	thr	Roc	NN	LDA	Net	Cmb
all	750	0.355	0.371	(0.384)	0.400	[0.397]
	2000	0.355	0.370	0.383	0.412	0.420
20	750	0.507	0.537	(0.562)	0.597	[0.595]
	2000	0.507	0.546	0.568	0.620	0.629

Table 1: Average precision at all relevant docs (all) and average precision at 20 docs (20) for various routing strategies. () - submitted xerox1, [] - corrected xerox2

As an alternative method of evaluation, we can rank the methods by their performance for each query. When these ranks are averaged, we get a measure that is less sensitive to extremely variable queries [1]. Table 2 presents the average rank for each method. The final column gives the rank difference that is statistically significant with a p-value of 0.05 according to the Friedman Test [13]. The test results should be taken with a grain of salt, since the Friedman Test assumes independence between methods (clearly violated here, since one method is a combination of the others!). However, it seems quite clear that the neural network and the combination run significantly outperform the other methods.

	thr	Roc	NN	LDA	Net	Cmb	sig.
all	750	2.08	2.76	2.74	3.58	3.84	0.51
	2000	2.19	2.59	2.68	3.48	4.06	0.53
20	750	2.40	2.61	2.86	3.58	3.55	0.45
	2000	2.27	2.64	2.76	3.60	3.73	0.48

Table 2: Average within-query rank of routing strategies.

It is valuable to look more closely at the choice of threshold. Previous research had found that average performance for individual classifiers was optimized by selecting a very restrictive threshold (750). The TREC-4

[1] See Hull [13] for reasons why this might be advisable.

results suggest that this is not always the case. Table 3 presents the fraction of queries which score better at a threshold of 2000 than at a threshold of 750 (queries with equal performance are ignored). For the individual classifiers, the less restrictive threshold hurts about as many (or more) queries as (than) it helps. However, the combination run is much more robust, allowing the system to apply advanced classification techniques to a much larger number of test documents, resulting in a corresponding improvement in performance.

	NN	LDA	Net	Cmb
all	0.383	0.425	0.511	0.652
20	0.500	0.458	0.588	0.733

Table 3: Percentage of queries where threshold 2000 is better than 750 (queries with equal performance are ignored).

Table 4 compares the performance of the Xerox routing system to other systems at TREC-4. In particular, it measures the absolute difference in uninterpolated average precision at all relevant documents between Xerox runs and the best result submitted to TREC-4 for each query. The submitted LDA run was within striking distance (5%) of the best performing system for 14 of the 50 queries and reasonably close (within 10%) for 27 of 50 queries. The best posthoc run (combination with threshold 2000) improves those numbers to 21 and 33 queries respectively. The system performs quite solidly but there is certainly room for improvement.

Another aspect that we would like to evaluate more closely is the effect of bias in the test set. The test set for TREC-4 consisted of the Ziff data from disk3 plus new Federal Registry documents. Since all of disk3 was part of the training set, this means that some queries were trained on data that was subsequently used to test performance. In particular, 6 out of the 25 Computer queries and 13 out of the 25 Federal registry queries had disk3 Ziff documents in the local region, and hence offer training sets potentially biased towards higher performance from our learning-intensive classifiers. Initial indications suggest that this is indeed the case for the biased Computer queries, but not the case for the Federal Registry queries.

	thr	$< 5\%$	$5-10\%$	$10-20\%$	$> 20\%$
LDA	750	14	13	15	8
Net	750	18	12	14	6
Cmb	2000	21	12	13	4

Table 4: Comparative results: absolute difference in uninterpolated average precision at all relevant documents (all) between method and best result submitted to TREC-4 for that query. Table values are number of queries within given range.

3 The Filtering Task

The filtering task is closely related to the routing task described in the previous section. The primary difference is that the system must also make a binary decision about whether to accept or reject each test document. Also, evaluation is based on a utility function of the form A*R-B*N where R and N are the number of relevant and non-relevant documents in the retrieved set and A and B are positive constants. The goal is to maximize this utility function. Three separate runs were submitted with utility functions (A,B) = (1,3), (1,1), and (3,1), which are optimized by selecting documents with an expected probability of relevance greater than p = .75, .5, and .25, respectively [17].

We estimate probabilities explicitly in the routing task as part of our combination strategy, as described in step 3 of the previous section. Therefore, to produce filtering results, we need merely filter the returned set according to these probability estimates. Note that only test documents which pass our initial thresholding step will be considered. In preliminary experiments, we found that a combination of the four classification techniques used in routing also worked well for filtering. However, we also discovered that the combination run tended to underestimate the actual probability of relevance, often quite substantially, and we obtained better performance by averaging the two largest probability scores for each document. Unfortunately, our filtering run submitted to NIST was also corrupted, since it was based on the same data used for the combination routing run. The results below have been generated from the corrected data.

P	thr	Roc	NN	LDA	Net	Cmb	Top2
75	750	2.64	3.30	3.66	4.20	3.43	3.77
50		2.50	3.00	3.47	3.75	4.07	4.21
25		2.61	2.87	3.52	3.62	4.17	4.21
75	2000	2.84	3.33	2.87	4.31	3.69	3.96
50		2.89	3.07	2.88	3.93	4.23	4.00
25		2.62	3.04	3.39	3.69	4.15	4.11

Table 5: Average within-query rank of filtering strategies.

Since the scale of the utility scores differs across queries, it is misleading to summarize the results simply by looking at the average utility. Instead, we compare classifiers using the average rank statistic, as presented in the previous section. The results are shown in Table 5. The new column Top2 is derived by taking the average of the largest two probability estimates for each document, and is designed to correct for the bias mentioned above. In general, the network and combination strategies tend to perform better than the alternatives, just as they did in routing. However, the combination run is much less effective (and the network run much more so) when the filter probability p = 0.75.

A filtering system must be capable of both ranking the documents accurately (the routing task) and selecting the proper size of the retrieved set. In order to compare between the routing and the filtering task, we attempt to separate performance due to ranking from performance due to selecting the right threshold. In order to measure this distinction, we take the ranked list returned by each classifier and compute the maximum utility which can be obtained from that ranking. Table 6 presents the average ranks for these optimal utility scores. By comparing the two tables, we learn that the combination run could score a lot better if it had more accurate probability estimates.

P	thr	Roc	NN	LDA	Net	Cmb	Top2
75		2.73	3.18	3.47	3.91	3.95	3.76
50	750	2.45	3.32	3.28	4.15	4.13	3.67
25		2.38	3.25	3.21	4.00	4.18	3.98
75		2.69	3.16	3.07	3.89	4.34	3.85
50	2000	2.48	2.74	3.09	4.10	4.54	4.05
25		2.25	2.71	3.02	3.97	4.79	4.26

Table 6: Average within-query rank of optimal filtering performance for each strategy.

In our preliminary experiments, we found that the probability estimates of the combination run often substantially underestimated the true probability of relevance. We obtained this information by comparing the observed number of documents in the relevant set to expected number, obtained by taking the sum of the probability estimates of the individual documents. Unfortunately, this method does not produce accurate summary statistics for the entire query set, because it does not include the queries where no documents are returned. An empty retrieved set often indicates that the system has underestimated the probability of relevance, which means that ignoring these queries may bias the average results. Therefore, we adopted a different approach. We computed the size of the optimal returned set for each query and method and compared it to the size of the actual returned set. Table 7 measures the percentage of the queries where the actual retrieved set is larger than the optimal retrieved set (ignoring queries where they are equal). Values much lower than 50% indicate that the technique is underestimating the probability of relevance for a large proportion of the queries and retrieving too few documents.

Table 7 reveals a number of interesting patterns. First, their is an overall tendency to return too few documents. Using the restrictive initial threshold of 750 is a large part of the problem. In hindsight, it is certainly a mistake to use a two-stage filtering algorithm and then expect the probability estimates used to filter in the second stage to be unbiased! The problem is most evident for thr = 750 and p=0.25, when many documents which are rejected in the first filtering stage have p>0.25 of being relevant.

P	thr	Roc	NN	LDA	Net	Cmb	Top2
75		0.418	0.316	0.591	0.290	0.186	0.318
50	750	0.396	0.326	0.532	0.370	0.302	0.500
25		0.245	0.245	0.327	0.367	0.327	0.449
75		0.418	0.461	0.682	0.333	0.186	0.452
50	2000	0.396	0.467	0.633	0.479	0.364	0.583
25		0.265	0.449	0.583	0.500	0.460	0.540

Table 7: Percentage of queries where actual retrieved set is larger than the optimal retrieved set (queries where they are equal are ignored).

When the initial threshold is increased to 2000, this bias is substantially reduced. It is also interesting to note the LDA behaves much differently from the other classifiers, tending to overestimate the size of the retrieved set.

We have already mentioned that the combination run performs much worse than expected for p = 0.75. From Table 7, it is very clear why. It is retrieving too few documents for over 80% of the queries. The problem comes from the fact that the combination run is constructed by taking the average of the probability estimates from four different classifiers, which means that its variance will be four times smaller. Since documents with p>0.75 are extreme in any event, far fewer documents are likely to satisfy this criterion for the combination run. Therefore, we can conclude that using the averaged probability estimates directly is not the right approach for filtering. Instead, we need to renormalize the estimates, perhaps by using logistic regression over the training set, in order to rescale the variance. Our naive initial attempt to correct for underestimation by using the top two probability estimates for each document seems to work reasonably well, but the optimal performance table tells us that the ranking produced by this measure is poorer, so we will be better off in the long run if we correct the probability estimates obtained for the original combination strategy.

4 Spanish Track

Our approach to Spanish is traditional from an information retrieval perspective. We are interested in testing whether our Spanish linguistic tools improve retrieval performance. For this work, we use SMART [1] as the underlying text retrieval system, but with all our linguistic analysis being done prior to indexing and retrieving documents. New fields are created for each document containing the different linguistic components we derive from the original text.

We use the following linguistic tools, developed at Xerox and Rank Xerox, to analyze the Spanish text:

- a morphological analyzer [15], which returns part of speech information and lemmatized forms of individual words (e.g. países → país +Noun+Masc+Pl)

```
<DOC> <DOCNO> SP94-0000662 </DOCNO>
<ARTNUM> 0000662 </ARTNUM>
<HEADLINE> San Marcos espera de NL arte joven            </HEADLINE>
<TEXT> La Subsecretaria de Cultura reune obras de artistas locales
       menores de 30 anos para enviarlas al certamen de la feria de
       Aguascalientes.                                  </TEXT>
<F1>   el subsecretaria de cultura reunir obra de artista local menor
       de 30 ano para enviar al certamen de el feria de Aguascalientes .
                                                        </F1>
<F2>   obra de artista local menor
       ano
       certamen
       feria de Aguascalientes                          </F2>
<F3>   reunir
       enviar                                           </F3>
<F4>   subsecretaria_de_cultura
       obra_de_artista_local_menor
       ano
       certamen
       feria_de_Aguascalientes                          </F4>  </DOC>
```

Figure 1: Sample Spanish document augmented with three additional fields: lemmatized words, noun phrases, lemmatized verbs, joined noun phrases. This format allows us to test different combinations of linguistically-derived information, without reindexing the corpus.

- a part-of-speech tagger [7], which uses the morphological analyzer and a trained hidden Markov model (HMM) network to choose the part-of-speech of a word from context
- a noun phrase extractor, which extracts noun phrase patterns from tagged text.

The major problem that we face in our Spanish experiments is dealing with accented text: some words that should have been accented were not, and we have had trouble getting SMART to recognize some accented characters. To solve the first problem, we modified our linguistic tools so that they correctly lemmatize unaccented words. For the second, we strip off accents before feeding the text to SMART. Fortunately, there are very few confusions between accented and unaccented words in Spanish.

Our treatment of a Spanish document begins by part of speech tagging the TEXT field contents. Each tagged word is then lemmatized according to its part of speech. From this tagged text, a number of supplementary fields are created and added to the original document: field F1 contains a lemmatized form of the text of field TEXT, field F2 contains all the lemmatized noun phrases of one word or more (one per line), field F3 contains all and only the lemmatized verbs, and field F4 contains all the lemmatized noun phrases with intervening spaces replaced by underscores so that they will be considered as units by SMART. Figure 1 gives a sample document. The total time needed to create these augmented documents over the 68,000 Spanish documents (200 MBytes) is 39 hours of real time on a SPARC 20. We treat the queries in a similar fashion.

We then conducted a preliminary analysis on the first 25 Spanish queries to determine which approaches were most successful. We submitted two runs, a baseline and one constructed using query expansion. The baseline runs uses the contents of fields F1 and F2, which corresponds to using lemmatized text and doubling the weight of the component terms in noun phrases. Indexing noun phrases by their components, rather than treating them as single units, produced slightly superior performance in the preliminary tests, which motivates this decision. Unfortunately, a programming error resulted in parts of some noun phrases being truncated from field F2, so the submitted run does not precisely match the desired experiment.

Since the the new Spanish queries are particularly short, there is reason to hope that they might benefit from some query expansion. Our approach is to take the first 20 documents returned by the baseline run and extract all terms that occur significantly more often in this document sample than one would expect by chance. The terms are selected according to a binomial likelihood ratio test [10], comparing their occurrence in the first 20 documents to their occurrence in the rest of the collection. The selected terms are then weighted in proportion to the significance of their occurrence in the sampled documents. Since it uses the baseline results, this run may also be affected by the programming error described above.

	query set	base	infl	infl-np	expand
all	Q1-25	0.454	0.484	0.492	0.467
	Q26-50	0.174	0.204	0.212	0.267
20	Q1-25	0.718	0.718	0.722	0.722
	Q26-50	0.306	0.354	0.378	0.402

Table 8: Average precision at all relevant docs (all) and average precision at 20 docs (20) for Spanish queries.

The corrected Spanish performance figures are presented in Table 8. We include four different runs: (1) base = stop list but no morphological analysis, (2) infl = text lemmatized (stemmed) with inflectional morphology, (3) infl-np = noun phrase weight doubled, and (4) expand = query expansion. The uncorrected performance figures for infl-np and expand on Q26-50 (corresponding to our submitted runs) are 0.190/0.366 and 0.238/0.380 respectively. We present separate results for queries 1-25 (used for TREC-3) and 26-50 (used for TREC-4) since the former are substantially longer, and we note that the results reflect the difference in length.

We find that lemmatization using inflectional morphology helps in most cases, making a 3-5% absolute difference in performance. However, when the queries are long and the user is examining fewer than 20 documents, there is no improvement. These conclusions agree with the results obtained for English [13], although the Spanish inflectional morphology is somewhat more effective than its English counterpart. Doubling the weight of noun phrases only slightly improves performance. Our query expansion technique is harmful for the long queries, but improves performance quite substantially for the short queries. Unfortunately, this improvement tends to be restricted to the queries where we are already doing well, so the value of this automatic expansion technique is limited.

When compared to other systems, the corrected infl-np run is more consistent, scoring above the median for all 25 queries, but always well below the best performing system. The corrected expansion run scores as well as the best system for 3 queries, but it is also below the median for 3 queries, as it tends to drag down performance for queries where no good expansion terms can be found. This suggests that we should look for an expansion technique that provides more consistent (if less dramatic) improvements in performance. In general, the Spanish linguistic tools provide solid though unspectacular benefits for the information retrieval problem.

5 Interactive Track

5.1 Goals of the Experiment

The interface used in this session represents the first time we have integrated Scatter/Gather [6] with TileBars [11] and Ranked Titles (standard similarity search via the vector space model). Users can display retrieval results in these three modes, each with a different ranking strategy, and the output of one mode can be used as input to another mode. The goal of the interface was to provide multiple ways for the users to view retrieval results, in the expectation that different modes and ranking orders are appropriate at different points in the search, and that the usefulness of a mode varies with the kind of query being investigated.

We predicted that subjects would use the clusters produced by Scatter/Gather to organize the initial retrieval results and select subsets of these results for further examination (or to eliminate subsets from consideration) and then use TileBars to help determine which documents are relevant to the query. We also suspected that users would make little use of the display of ranked titles given the TileBar visualization and ranking as an alternative.

We originally planned to run experiments that would test each interface mode individually, and to examine the effectiveness of each mode on particular queries, but this would have required more subject hours than could be accommodated in the time available.

5.2 The System

Our system consists of the TDB [8] (Text DataBase) system developed at PARC and a new user interface that combines standard vector space search, Scatter/Gather, and the TileBar display method. TDB is implemented in Common LISP and CLOS, and the interface is implemented in TCL/TK [19]. The two parts communicate with one another through ILU [5] and expectk [18].

A flow diagram of the process model for the Interactive Track Interface is shown in Figure 2. First the user specifies a query, (in the form of a list of topics, see below). A threshold k is set indicating how many documents are to be retrieved initially. Then the k top-ranked documents, according to the vector space model, are retrieved and shown to the user in Title Mode. After this, the user can switch the display of the retrieval results among the three modes of Titles, TileBars, and Scatter/Gather. The user can view a subset of the retrieval results by selecting one or more of the clusters produced by Scatter/Gather, thus indicating that only the contents of those clusters are to be viewed. (The system keeps track of state information and allows the user to back up to previous states.) The user can reformulate the description of the query if desired. At all points, document titles can be marked as relevant. At the end of the session, the documents so marked are saved into a file.

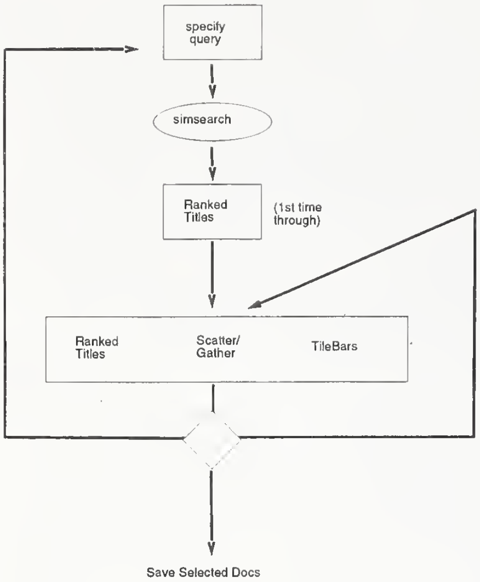


Figure 2: A flow diagram of the process model for the Interactive Track Interface.

As stated above, TDB provides a standard vector space weighting and ranking scheme, similar to that reported in [2] as well as standard Boolean search. It also includes support for Scatter/Gather clustering, as well as something we call structured similarity search, which provides support for TileBars. Structured similarity search works the same way as standard similarity search (the vector space model), except that it returns a list of term offsets that correspond to the query term sets as described below.

Our system as shown to users did not include relevance feedback or query expansion via suggestions of related terms although these mechanisms fit within our framework and will be included in future. Relevance feedback would most likely have improved the results, but we had not yet incorporated this mechanism with the structured similarity search at the time the experiments were run.

5.3 The User Interface

Our interface focuses on helping users understand and explore their retrieval results, rather than concentrating on query formulation and refinement. This is not to say that query reformulation is unimportant, but rather that we chose not to emphasize that for these experiments. Furthermore, we have evidence that the Scatter/Gather interface helps users determine alternative terms by which to augment their queries, see [20].

Figure 3 shows the entire interface when set in TileBar mode. We will use query 216 as a running example with which to illustrate the components of the interface:

> *What research is ongoing to reduce the effects of osteoporosis in existing patients as well as prevent the disease occurring in those unafflicted at this time?*

The lefthand side of the interface contains the user's query specification, the system's translation of the query, an active log of the state of the system as the user switches from mode to mode, and a window showing the documents the user has saved for the query. There is also a radiobutton that allows the user to specify the document cutoff threshold for the vector space search. The righthand side of the interface consists of a set of buttons that allow the user to change the mode of the retrieval results display, the display of the retrieval results, and a window for displaying document contents.

In Figure 3 the user has selected two documents as relevant, as indicated by the dark circles, and their titles appear in the "Saved Relevant Documents" window.

5.3.1 Specifying the Query

To accommodate the TileBar interface, users are required to enter their queries into a sequence of entry windows, as shown in the upper lefthand corner of the interface. The user is told that each line should correspond to a different topic of the query. Each entry line is called a "termset" since it is meant to contain a set of terms representing one topic. In this example, the participant has broken query 216 into four topics (in this case there is only one word per topic): *osteoperosis, patient, prevention,* and *research.*

In other recent work [12], we have found that we can achieve very strong improvements in precision at high document cutoff levels by first requiring the query to be specified in terms of a list of topics, and then applying the following two constraints:

- Treat the list of topics as a Boolean conjunct of disjuncts,
- Apply a proximity constraint of 100 tokens to this conjunct of disjuncts

In other words, the terms in each topic are treated as a disjunction, and a conjunction is imposed among the topical disjuncts. Only the documents which pass through the filter of these two constraints are retained, and they are ranked according to the vector space ranking, as before. The topmost termset is considered most important, and termsets lower down in the list are considered less important (some variations of this algorithm in which not all termsets are required have been experimented with). This algorithm yielded very strong improvements over using the short query with no constaints, or on using the original query specification, in the case of TREC 4 queries.

Similar results were found independently by the Waterloo group at TREC 4 [4]. In that work, the queries were again specified as an ordered list of topics, called subqueries. Most subqueries were specified as disjuncts. Each subquery was ranked separately, and then the ranks were combined by using the ranking of the topmost first, the secondmost second, and so on. The proximity constraint was slightly different in this algorithm – instead of being fixed, the documents were ranked according to the inverse of the distance between pairs of subqueries. The results obtained for the manual ad hoc using this approach were quite strong.

In this set of experiments, however, we did not treat the query as a Boolean query, but rather converted it into a bag of words and performed a standard similarity search on the bag. The separate lines were made use of, however, in order to display the corresponding hits in the TileBar display, described below.

5.3.2 Retrieval Result Modes

Figure 4 shows the initial results of the query as displayed in Title Mode, shown in ranked order according to the vector space model. The rank of the document is shown along side its title. When the document had no preassigned title then its document id label is shown instead. In this example, many of the top-ranked titles are very uninformative.

Figure 5 shows the results of running Scatter / Gather on the 250 retrieved documents. The results are partitioned into four clusters, of the sizes shown. Associated with each cluster is a list of "topical terms," extracted from the cluster centroids. These terms are meant to indicate the central topics encompassed by the documents within the cluster. In the figure we can see that the topical terms of Cluster 1, e.g., *bone osteoporosis, fracture, estrogen*, indicate that it should contain relevant documents. The topical terms for Clusters 3 and 4 make them appear to contain non relevant documents, whereas Cluster 2 may contain some relevant documents as it discusses *aid, cancer, institute, heart, center*, etc.

When the user selects Cluster 1 and then switches to TileBar mode, the view of Figure 3 results. The subset of 19 documents from Cluster 1 appear in the TileBar display. The user can use the Backup button to go back to the previous state, as shown in the History window. The TileBar representation works as follows. Each rectangle represents a document. Each row of the rectangle represents the corresponding termset in the query display, i.e, the top row corresponds to *osteoporosis*, the second row to *patient*, etc. Each row of each rectangle is comprised of a sequence of squares. Each square indicates a segment of the document; the leftmost square indicates the first segment, or paragraph, or other unit, of the document, the square to the right of this indicates the second segment of the document, and so on. The darkness of the square corresponds to the number of times the query occurs in that segment of text; the darker the square the greatei the number of hits. White indicates no hits on the query term. Thus the user can quickly see if some subset of the terms overlap in the same segment of the document.

The TileBar ranking order is different than that of the vector space method. In this implementation, documents are first ranked by number of segments in which hits for all termsets overlap, second by the total number of hits in the document, and third by the vector space ranking.

As can be seen in Figure 3, the terms of the query are highlighted in the document display window; each color corresponds to a different line of the user query. In Figure 3 the contents of document FR88513-0157 appear in the appropriate window. This document was ranked highest among the 19 documents according to the TileBar ranking strategy. The portion of the document shown is one where the term hits overlap. A sentence is visible that states "Research is revealing that prevention may be achieved through estrogen replacement therapy for older women ..." and the rest of the context indicates that thing to be prevented is osteoporosis. Unfortunately, the TREC judges did not mark this document as relevant, perhaps because the article also contains a discussion of "National Osteoporosis Prevention Week". However, the TileBars strongly indicate that research is discussed in the same context as osteopororis and prevention in this document; perhaps use of such a tool could aid in the relevance judging process.

5.3.3 Interface Design Issues

We chose to lay out the components of the interface in one large window, in an attempt to keep in view all pertinent information at all times. This is as opposed to making use of menus and pop-up windows. We also wanted keep the number of possible mouse operations small. This information-constancy effect, although difficult to achieve due to limited screen real estate, acted as a useful constraint on the design of the interface.

As an example of the usefulness of this constraint, we decided that we wanted a clear, simple way for the user to select relevant documents. We decided to place a "checkbox" next to each document rather than requiring the users to perform cut-and-paste operations, or requiring them to remember which mouse button corresponded to selecting a document as relevant (clicking on a document title brings up the document for display rather than selecting that document as relevant). Several somewhat unexpected benefits resulted from this design choice. First, a marked checkbox is very distinctive visually. Second, the marks give an impression of the relative positions of the relevant and nonrelevant documents when a subset is reranked as the result of switching display modes. Third, the salience of the marked boxes potentially helps the user maintain some understanding the state of the search – more dark spots indicating more success with the search. Finally, when used in conjunction with the cluster display, the marked boxes sometimes served to signal a cluster that is interesting by virtue of the fact that it contains many relevant documents. One participant in our study remarked: "Sometimes I'd just go through the titles and select the ones that obviously pertained to the query, and then I would look at the clusters and see if they were concentrated at all. And sometimes that would happen and sometimes not."

Our participant interviews indicated that some of the participants did not like this all-in-one layout. This opinion may be caused by factors not directly related to usability, e.g., it may have been voiced because the participants were used to software that makes use of small pop-up windows, and because the fonts were too small (a comment given by some of the participants).

5.4 Experimental Design

Good experimental design requires that many different participants for each (query, system) combination, in order to reduce the effects of individual variation. Unfortunately, because each query requires 30 minutes to run, and because the rules of the track require that all 25 queries be covered, time constraints limited the amount of replication possible.

We attempted to maximize the experimental validity of our results while at the same time meeting the requirement of having users complete all 25 TREC queries. Our study consisted of four UC Berkeley graduate students, each of whom executed 13 queries. These consisted of 12 of the required 25, as well as one extra query given to all four participants. Only two of the participants' results on this query were reported, chosen arbitrarily.

We evaluated the queries in advance in order to rate them according to expected difficulty in general and with each search mode type. This prediction was handicapped in that it made use of restricted earlier versions of the interfaces and was indexed over only a subset of the collection used in TREC 4. We used these predictions for the experimental design, in order to ensure that each participant received a mix of "interface favorable" and "interface unfavorable" queries. These classifications can also be read as "easy" vs. "difficult".

Given these constraints, we developed a nested-factorial design to maximize within-participant measurement. In this design we nested, or split, queries across pairs of participants. Queries were assigned to participants as shown in Table 9. Participants were exposed initially to easy queries in each session, and these were then followed by harder ones.

Participants completed queries in two sessions. The experiments were run in an otherwise empty room with a video camera recording the session. Participants were given an 10-minute demonstration of the interface followed by a 10-minute warmup exersize, and the participants were provided with a 3 page description of the interface for reference. Additionally, a binder of topic descriptions was prepared for each participant, with each topic description appeared on the top of a separate page. Participants were not allowed to look at a new topic before the current one was completed.

Only 30 minutes was allowed per query, as specified in the instructions for the TREC interactive track. Participants were allowed to take short breaks between queries if desired. The instructions for the task were given as in the interactive track specifications "find as many good documents as you can for a topic, in around 30 minutes, without collecting too much rubbish." We took this as a hard time limit, and participants were required to stop when the 30 minute time limit was up. This statement emphasizes the finding of many relevant documents and deemphasizes the undesirability of including nonrelevant documents, and this had ramifications for how the participants performed. Some participants saved large numbers of documents for some of the queries without checking carefully for relevance, thus lowering overall precision.

We logged a good portion of the participants' activity, including which search state and mode type (Scatter/Gather, Tilebars, Titles) was in use when an action took place. We chose not to record every mouse event but did record when the search window or the document viewer windows were scrolled, as well as when clusters were selected and when documents were saved (and unsaved) as relevant. Since we recorded the visible contents of the search windows, we can make inferences about what documents were in view and what actions the user took in response to this information.

5.5 Results

5.5.1 Precision and Recall

The per-query results for precision and recall are shown in Tables 10 and 11. Our participants performed strikingly well on those queries with relatively few possible relevant documents that we predicted the interface would be helpful with, e.g., 207 and 216, and did poorly in the converse, e.g. 243, 236, 232, and 208. The relation between predicted difficulty and actual performance are more difficult to interpret for some queries, such as 220 and 216, but our system appears to have done as well as or better than other systems on these queries. Overall, the predictions

	A			B			A	A			B		
S1	210	202	207	203	204	205	212	211	216	215	206	208	209
S2	213	220	227	223	232	236	212	214	242	250	238	239	243
S3	210	202	207	203	204	205	212	211	216	215	206	208	209
S4	213	220	227	223	232	236	212	214	242	250	238	239	243

Table 9: Experimental Design. Queries were classified in advance as "interface favorable" (A) or "interface unfavorable" (B) and each participant was given a mix of the two types in the order shown, left to right. Four participants were used and each searched on half of the required queries.

Topic	Ret	Rel	RR	Prec.	Recall
202	19	283	13	0.684	0.045
203	6	33	1	0.166	0.030
204	5	397	3	0.600	0.007
205	4	310	1	0.250	0.003
206	2	47	2	1.000	0.042
207	67	74	41	0.611	0.554
208	6	54	2	0.333	0.037
209	16	87	8	0.500	0.091
210	36	57	27	0.750	0.473
211	26	323	25	0.961	0.077
212	21	153	18	0.857	0.117
213	12	21	5	0.416	0.238
214	3	5	3	1.000	0.600
215	26	183	23	0.884	0.125
216	24	36	17	0.708	0.472
220	10	24	5	0.500	0.208
223	26	363	14	0.538	0.038
227	85	347	71	0.835	0.204
232	1	9	0	0.000	0.000
236	14	43	0	0.000	0.000
238	48	270	28	0.583	0.103
239	100	123	20	0.200	0.162
242	11	38	6	0.545	0.157
243	15	69	1	0.066	0.014
250	25	86	10	0.400	0.116
TOTS/	608	3435	344	0.5658/	0.1001/
AVGS				0.5357	0.1570

Table 10: Scores determined by NIST for run XERINT1. Note that this represents the results of two different participants, paired arbitrarily. Ret = number retrieved, Rel = number possible relevant, RR = number retrieved that were relevant, Prec. = precision. Both macro and micro averages are shown for precision and recall.

Topic	Ret	Rel	RR	Prec.	Recall
202	39	283	12	0.307	0.042
203	15	33	3	0.200	0.090
204	5	397	4	0.800	0.010
205	1	310	1	1.000	0.003
206	10	47	7	0.700	0.148
207	44	74	32	0.727	0.432
208	4	54	2	0.500	0.037
209	24	87	13	0.541	0.149
210	33	57	27	0.818	0.473
211	22	323	19	0.863	0.058
212	23	153	11	0.478	0.071
213	19	21	7	0.368	0.333
214	4	5	3	0.750	0.600
215	42	183	36	0.857	0.196
216	29	36	13	0.448	0.361
220	15	24	11	0.733	0.458
223	37	363	2	0.054	0.005
227	52	347	46	0.884	0.132
232	3	9	2	0.666	0.222
236	41	43	6	0.146	0.139
238	55	270	30	0.545	0.111
239	20	123	8	0.400	0.065
242	10	38	9	0.900	0.236
243	16	69	1	0.062	0.014
250	22	86	7	0.318	0.081
TOTS/	585	3435	312	0.5333/	0.0908/
AVGS	585	3435	312	0.5629	0.1791

Table 11: Scores determined by NIST for run XERINT2. Note that this represents the results of two different participants, paired arbitrarily. Ret = number retrieved, Rel = number possible relevant, RR = number retrieved that were relevant, Prec. = precision. Both macro and micro averages are shown for precision and recall.

about the easy vs. difficult queries were born out. The average precision and recall scores for A vs. B queries were as follows:

A1: Prec .65 Rec .25

A2: Prec .61 Rec .33

B1: Prec .38 Rec .04

B2: Prec .50 Rec .08

What remains to be determined is whether or not other systems found the same queries easy and difficult in order to help determine if this effect is a function of the query, the interface, or both.

5.5.2 Participant Interviews

After the sessions the participants were interviewed about the use of the interface, and the results of these interviews were recorded and transcribed. The answers to the questions were quite informative. In answer to "What did you like best about the interface?" the participants answered as follows.

All four participants said explicitly that they liked the TileBar Interface or found it to be useful. One said: "I really like the tilebars the best. That's something unique. You can just click on it and get to that part of the document, and that's nice. It's like a magnifying glass." Another participant, while finding the TileBars useful, pointed out a problem with them, that sometimes even if the terms overlap, they do not necessarily overlap in a useful way and the visualization does not distinguish these two cases.

One participant had an interesting comment to make about the format in which queries were entered, saying: "I think having the four term sets is very useful. It was limiting, but on the other hand it really makes you think of the most important terms."

One participant was especially enthusiastic toward the clustering, finding the clusters useful for weeding out non-relevant documents, but did express concern about tossing out appropriate documents. Two participants mentioned liking the "sticky" checkboxes for selected relevant documents even after multiple searches on the same query. One mentioned the usefulness of the multi-color term highlighting in the display documents, where each color corresponds to a different query termset.

The participants were also asked "Is there anything [else] you didn't like about the interface?" In answer to this question and some of the others, we learned that system performance was one of the biggest problems. All four participants said that if the search performance had been better they would have done more searches. Two participants thought the sizes of the TileBars and the titles should be larger. One wanted keyboard accelerators. One participant was frustrated by the lack of a search abort capability. Two participants wanted a NOT operator and a phrase specification facility. One other participant asked for explicit AND and OR operators. All four thought that a term suggestion facility might have helped, but to differing degrees.

When asked how and when they used the TileBar facility, the participants answered as follows (these answers reveal information about the search strategies in general):

> "Basically, I used it after I had narrowed down the search a bit with the clusters. I usually used it to select relevant articles. I found the tile conjunctions useful to find phrases like "rain forest" but they weren't perfect. The tiles weren't always helpful if you had a fairly common word, or if it could be used in another way."

> "Almost all the time. It is the quickest way to tell which are the documents that have the most key words in it. If you get bored and don't want to read anymore, that's the quickest way to go."

> "That was usually a final part of the task. I would usually do the keyword selection, then I would do a clustering either once or twice depending on the results. Then I would go to the tilebar last and sometimes I would go to the tilebar mode, but not use it. I would just start scanning the titles and occasionally look at the tilebars. Other times I would really heavily use the tilebars. It just depended on the nature of the search."

> " ... I usually used it at the end. So once I got down to under 30, or around 30, documents then I'd just go look at the tilebars. ... I was looking for the conjunction of rain and forest and climate. Because not very many articles had all three together. But usually I'd wait until I only had a small clustering."

When asked how and when they used the Scatter/Gather display, the participants said they mainly used them to narrow down the set of articles to be viewed with TileBars and to eliminate unpromising documents. Large clusters were often reclustered. None of the participants thought having more than five clusters would be a good idea. Some users interwove the use of Scatter/Gather and TileBars.

When asked how and when they used the title display, all four participants said they didn't use it much because the other methods were more descriptive. Additionally, two participants said that the rankings were unhelpful and often misleading.

5.6 Conclusions

We find these preliminary results to be very encouraging; the participants performed well compared to the initial results returned for the other systems, were able to learn to use the new interface modes with very little acquaintance, and were enthusiastic about the new modes. The results of the participant surveys, a portion of which is reported in the preceding section, have given us very useful feedback about the merits of the interface and how it can be improved. On the top of the list is to add a term expansion suggestion mechanism, relevance feedback, improve the representation of the cluster contents, and improve search time performance.

We are currently devising measures to assess the usefulness of the display modes in various situations, based on choices users made given how many relevant documents were in view. We would also like to have available detailed transcripts of users of other systems, in order to help understand which kinds of displays are most helpful. Finally, we may conduct experiments in which only one of the three modes is available to facilitate evaluation of the effectiveness of each mode type.

Acknowledgements

We would like to acknowledge the efforts of Christine Diehl who conducted the participant studies, helped create the materials that were shown to the participants, helped determine the query difficulty rankings, transcribed the subject interviews and gave helpful comments on the design of the interface. Patricia Schank, now of SRI, also helped with the design of the experiment, determination of query difficulty, and commented on the design of the interface.

References

[1] Chris Buckley. Implementation of the smart information retrieval system. Technical Report 85-686, Cornell University, 1985.

[2] Chris Buckley, James Allan, and Gerard Salton. Automatic routing and ad-hoc retrieval using SMART: TREC 2. In Donna Harman, editor, *Proceedings of the Second Text Retrieval Conference TREC-2*. National Institute of Standards and Technology Special Publication 500-215, 1994.

[3] Chris Buckley, Gerard Salton, and James Allan. The effect of adding relevance information in a relevance feedback environment. In *Proc. 17th Int'l Conference on R&D in IR (SIGIR)*, pages 292–300, 1994.

[4] Charles L. A. Clarke, Grodon V. Cormack, and Forbes J. Burkowski. Shortest substring ranking (multitext experiments for TREC-4. In Donna Harman, editor, *Proceedings of the Fourth Text Retrieval Conference TREC-4*. National Institute of Standards and Technology Special Publication, 1996. (this volume).

[5] A. Courtney, W. Janssen, D. Severson, M. Spreitzer, and F. Wymor. *Inter-Language Unification, release 1.5*. Xerox PARC, 1994. ftp://ftp.parc.xerox.com/pub/ilu/ilu.html.

[6] D. R. Cutting, D. R. Karger, J. O. Pedersen, and J. W. Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In *Proc. 15th Annual Int'l ACM SIGIR Conference on R&D in IR*, June 1992. Also available as Xerox PARC technical report SSL-92-02.

[7] Doug Cutting, Julian Kupiec, Jan Pedersen, and Penelope Sibun. A practical part-of-speech tagger. *Proc. of the Third Conference on Applied Natural Language Processing*, April 1992.

[8] Douglass R. Cutting, Jan O. Pedersen, and Per-Kristian Halvorsen. An object-oriented architecture for text retrieval. In *Conference Proceedings of RIAO'91, Intelligent Text and Image Handling, Barcelona, Spain*, pages 285–298, April 1991. Also available as Xerox PARC technical report SSL-90-83.

[9] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

[10] Ted Dunning. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74, 1993.

[11] Marti A. Hearst. Tilebars: Visualization of term distribution information in full text information access. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*, Denver, CO, May 1995. ACM.

[12] Marti A. Hearst. Improving full-text precision using simple query constraints. In *Proceedings of the Fifth Annual Symposium on Document Analysis and Information Retrieval (SDAIR)*, Las Vegas, NV, April 1996. (to appear).

[13] David Hull. Stemming algorithms - a case study for detailed evaluation. *Journal of the American Society for Information Science*, 1996. To appear in special issue on the evaluation of IR systems.

[14] Ronald M. Kaplan and Martin Kay. Regular models of phonological rule systems. *Computational Linguistics*, 20(3):331–378, September 1994.

[15] Lauri Karttunen. Constructing lexical transducers. In *Proc. of the 15th International Conference on Computational Linguistics, COLING'94*, Kyoto, Japan, 1994.

[16] K.L. Kwok, L. Grunfeld, and D.D. Lewis. Trec-3 adhoc, routing retrieval and thresholding experiments using pircs. In *Proceedings of the 3nd Text Retrieval Conference (TREC-3)*, pages 247–255, 1995.

[17] David Lewis. Evaluating and optimizing autonomous text classification systems. In *Proc. 18th Annual Int'l ACM SIGIR Conference on R&D in IR*, pages 246–255, 1995.

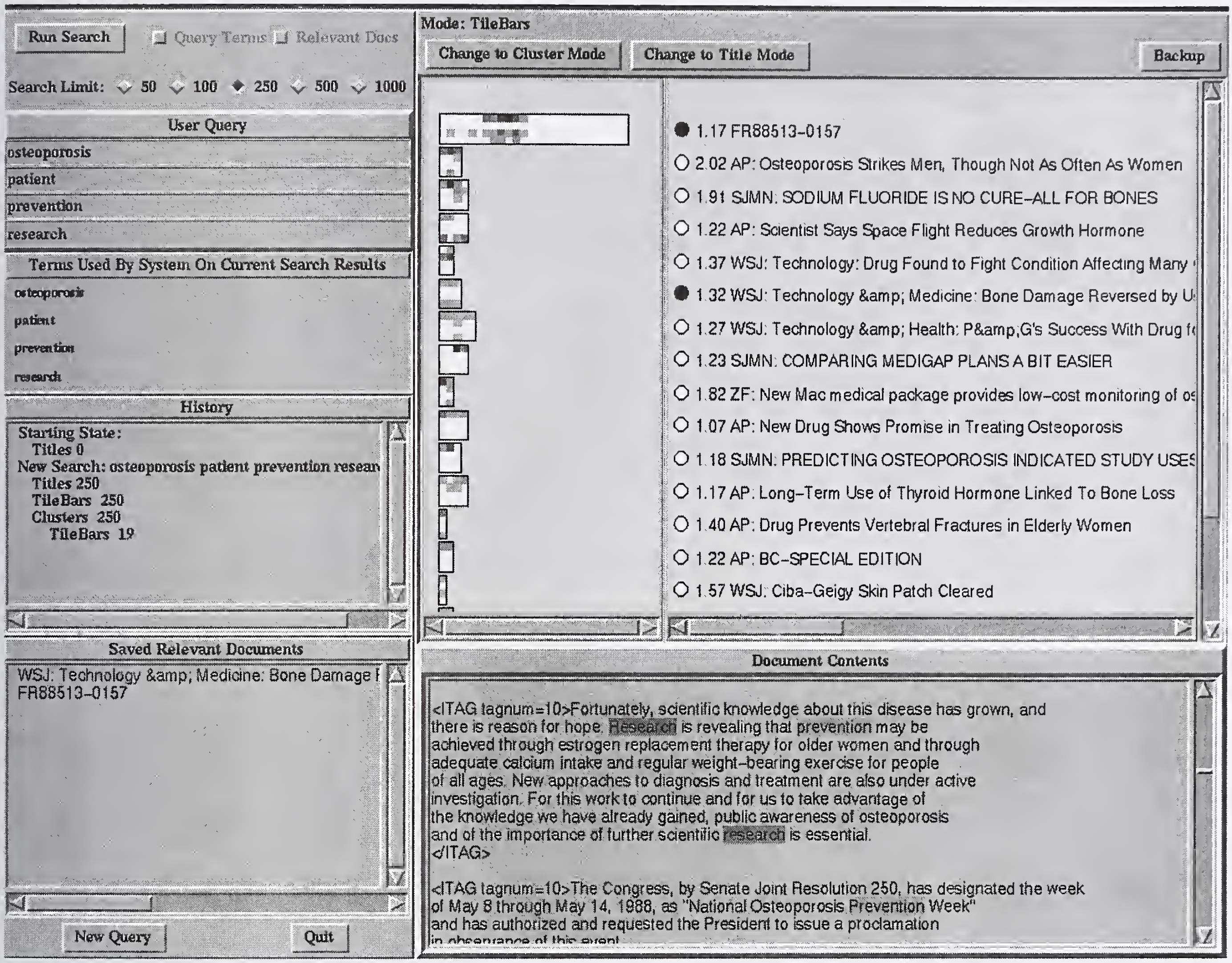
[18] Don Libes. expect: Curing those uncontrollable fits of interaction. In *Proceedings of the Summer 1990 USENIX Conference*, Anaheim, CA, June 1990.

[19] John Ousterhout. An X11 toolkit based on the Tcl language. In *Proceedings of the Winter 1991 USENIX Conference*, pages 105–115, Dallas, TX, 1991.

[20] Peter Pirolli, Patricia Schank, Marti A. Hearst, and Christine Diehl. Scatter/gather browsing communicates the topic structure of a very large text collection. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*, Vancouver, WA, May 1996. ACM. (to appear).

[21] Hinrich Schütze, David Hull, and Jan Pedersen. A comparison of document representations and classifiers for the routing problem. In *Proc. 18th Int'l Conference on R&D in IR (SIGIR)*, pages 229–237, 1995.

[22] Yiming Yang. Expert network: Effective and efficient learning form human decisions in text categorization and retrieval. In *Proc. 17th Int'l Conference on R&D in IR (SIGIR)*, pages 13–22, 1994.



Mode: TileBars

Search Limit: ◇ 50 ◇ 100 ◆ 250 ◇ 500 ◇ 1000

User Query

osteoporosis

patient

prevention

research

Terms Used By System On Current Search Results

osteoporosis

patient

prevention

research

History

Starting State:
  Titles 0
New Search: osteoporosis patient prevention resear
  Titles 250
  TileBars 250
  Clusters 250
    TileBars 19

Saved Relevant Documents

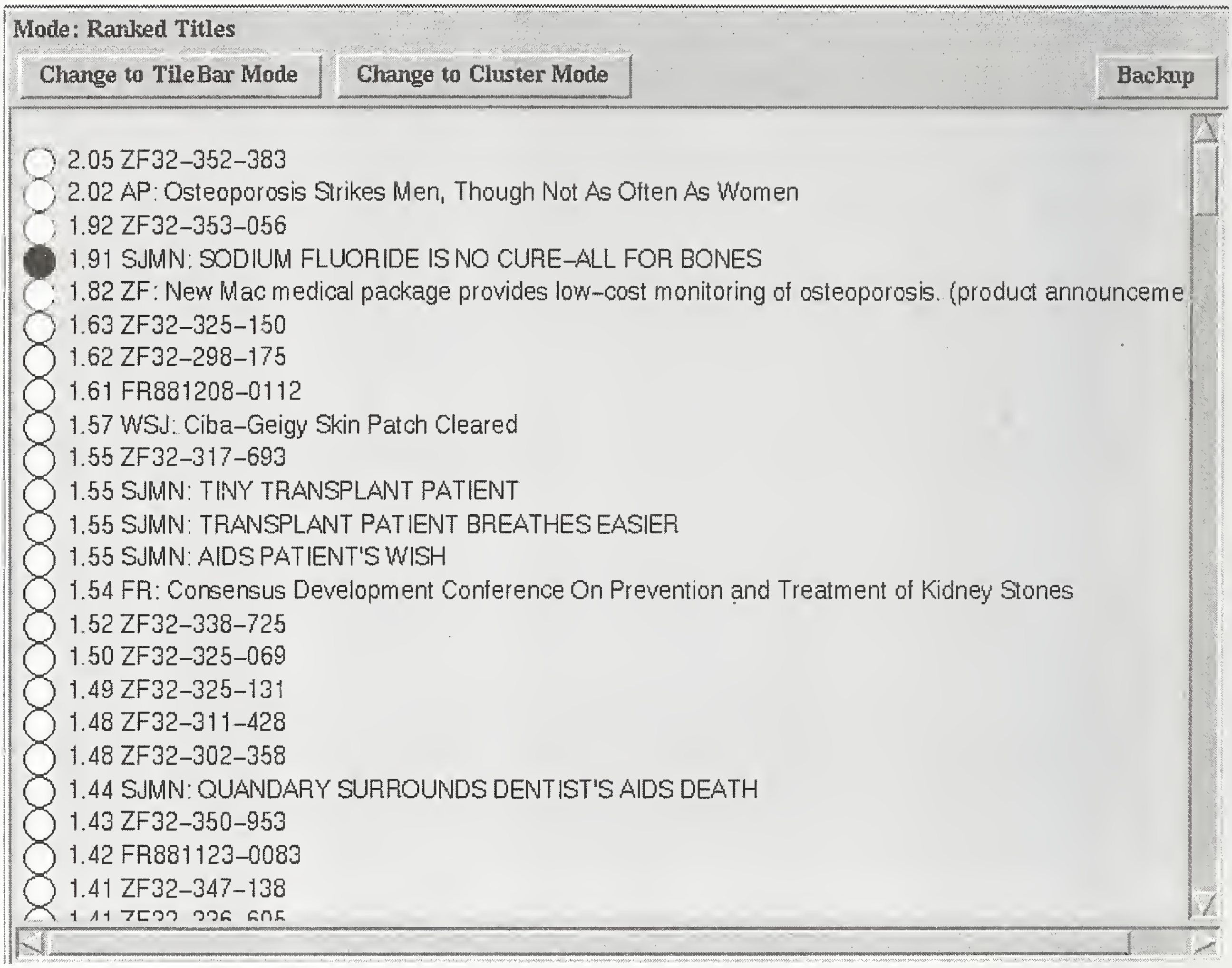
WSJ: Technology &amp; Medicine: Bone Damage I
FR88513-0157

- ● 1.17 FR88513-0157
- ○ 2.02 AP: Osteoporosis Strikes Men, Though Not As Often As Women
- ○ 1.91 SJMN: SODIUM FLUORIDE IS NO CURE-ALL FOR BONES
- ○ 1.22 AP: Scientist Says Space Flight Reduces Growth Hormone
- ○ 1.37 WSJ: Technology: Drug Found to Fight Condition Affecting Many
- ● 1.32 WSJ: Technology &amp; Medicine: Bone Damage Reversed by U
- ○ 1.27 WSJ: Technology &amp; Health: P&amp;G's Success With Drug f
- ○ 1.23 SJMN: COMPARING MEDIGAP PLANS A BIT EASIER
- ○ 1.82 ZF: New Mac medical package provides low-cost monitoring of os
- ○ 1.07 AP: New Drug Shows Promise in Treating Osteoporosis
- ○ 1.18 SJMN: PREDICTING OSTEOPOROSIS INDICATED STUDY USES
- ○ 1.17 AP: Long-Term Use of Thyroid Hormone Linked To Bone Loss
- ○ 1.40 AP: Drug Prevents Vertebral Fractures in Elderly Women
- ○ 1.22 AP: BC-SPECIAL EDITION
- ○ 1.57 WSJ: Ciba-Geigy Skin Patch Cleared

Document Contents

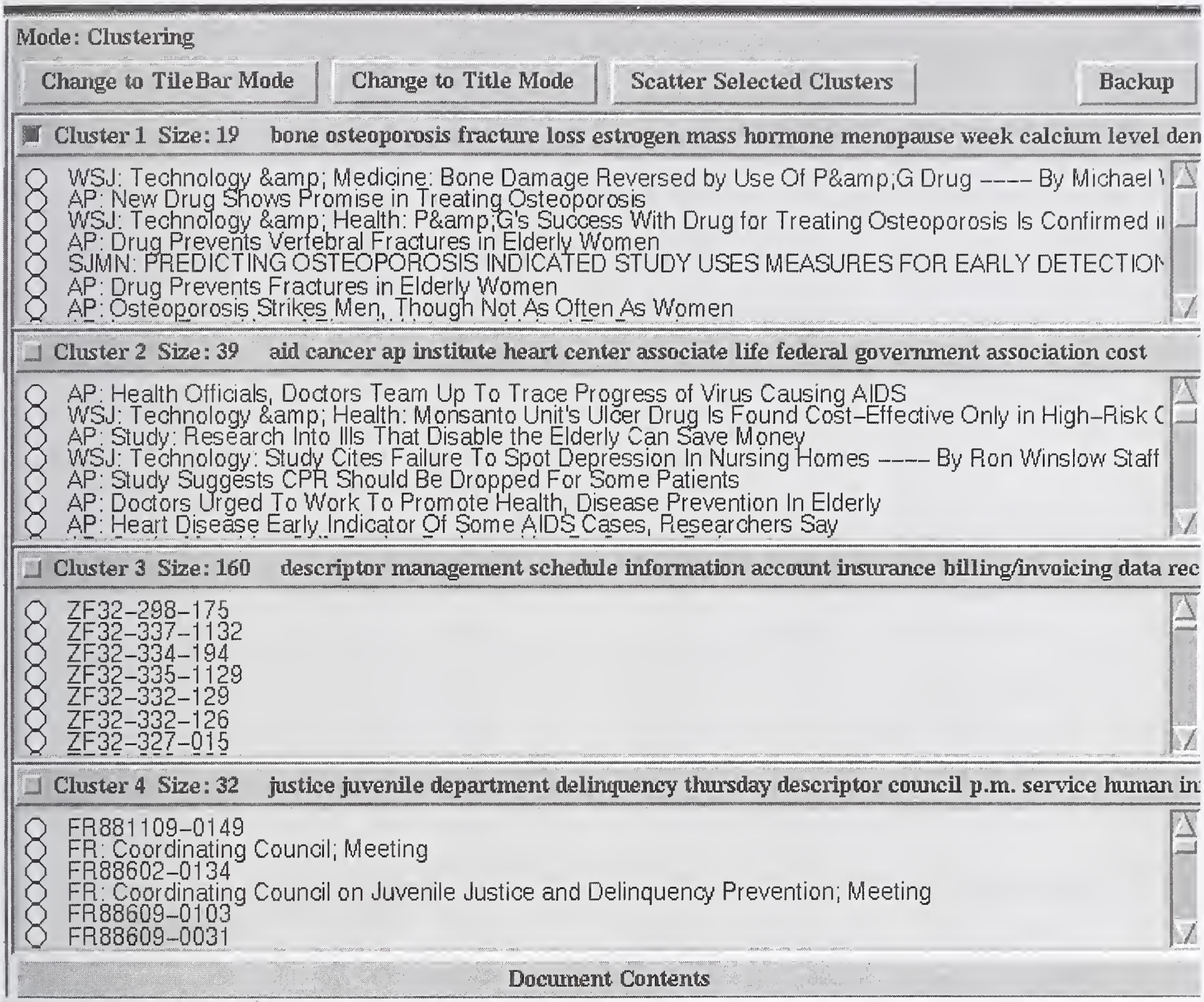
```
<ITAG tagnum=10>Fortunately, scientific knowledge about this disease has grown, and
there is reason for hope. Research is revealing that prevention may be
achieved through estrogen replacement therapy for older women and through
adequate calcium intake and regular weight-bearing exercise for people
of all ages. New approaches to diagnosis and treatment are also under active
investigation. For this work to continue and for us to take advantage of
the knowledge we have already gained, public awareness of osteoporosis
and of the importance of further scientific research is essential.
</ITAG>

<ITAG tagnum=10>The Congress, by Senate Joint Resolution 250, has designated the week
of May 8 through May 14, 1988, as "National Osteoporosis Prevention Week"
and has authorized and requested the President to issue a proclamation
in observance of this event
```

Figure 3: The PARC TREC 4 Interactive Interface, in TileBar mode.



○ 2.05 ZF32–352–383
○ 2.02 AP: Osteoporosis Strikes Men, Though Not As Often As Women
○ 1.92 ZF32–353–056
● 1.91 SJMN: SODIUM FLUORIDE IS NO CURE–ALL FOR BONES
○ 1.82 ZF: New Mac medical package provides low–cost monitoring of osteoporosis. (product announceme
○ 1.63 ZF32–325–150
○ 1.62 ZF32–298–175
○ 1.61 FR881208–0112
○ 1.57 WSJ: Ciba–Geigy Skin Patch Cleared
○ 1.55 ZF32–317–693
○ 1.55 SJMN: TINY TRANSPLANT PATIENT
○ 1.55 SJMN: TRANSPLANT PATIENT BREATHES EASIER
○ 1.55 SJMN: AIDS PATIENT'S WISH
○ 1.54 FR: Consensus Development Conference On Prevention and Treatment of Kidney Stones
○ 1.52 ZF32–338–725
○ 1.50 ZF32–325–069
○ 1.49 ZF32–325–131
○ 1.48 ZF32–311–428
○ 1.48 ZF32–302–358
○ 1.44 SJMN: QUANDARY SURROUNDS DENTIST'S AIDS DEATH
○ 1.43 ZF32–350–953
○ 1.42 FR881123–0083
○ 1.41 ZF32–347–138
○ 1.41 ZF32–336–605

Figure 4: The interface in Title mode.



■ Cluster 1 Size: 19 bone osteoporosis fracture loss estrogen mass hormone menopause week calcium level den

○ WSJ: Technology &amp; Medicine: Bone Damage Reversed by Use Of P&amp;G Drug ---- By Michael \
○ AP: New Drug Shows Promise in Treating Osteoporosis
○ WSJ: Technology &amp; Health: P&amp;G's Success With Drug for Treating Osteoporosis Is Confirmed i
○ AP: Drug Prevents Vertebral Fractures in Elderly Women
○ SJMN: PREDICTING OSTEOPOROSIS INDICATED STUDY USES MEASURES FOR EARLY DETECTIOI
○ AP: Drug Prevents Fractures in Elderly Women
○ AP: Osteoporosis Strikes Men, Though Not As Often As Women

□ Cluster 2 Size: 39 aid cancer ap institute heart center associate life federal government association cost

○ AP: Health Officials, Doctors Team Up To Trace Progress of Virus Causing AIDS
○ WSJ: Technology &amp; Health: Monsanto Unit's Ulcer Drug Is Found Cost-Effective Only in High-Risk C
○ AP: Study: Research Into Ills That Disable the Elderly Can Save Money
○ WSJ: Technology: Study Cites Failure To Spot Depression In Nursing Homes ---- By Ron Winslow Staff
○ AP: Study Suggests CPR Should Be Dropped For Some Patients
○ AP: Doctors Urged To Work To Promote Health, Disease Prevention In Elderly
○ AP: Heart Disease Early Indicator Of Some AIDS Cases, Researchers Say

□ Cluster 3 Size: 160 descriptor management schedule information account insurance billing/invoicing data rec

○ ZF32-298-175
○ ZF32-337-1132
○ ZF32-334-194
○ ZF32-335-1129
○ ZF32-332-129
○ ZF32-332-126
○ ZF32-327-015

□ Cluster 4 Size: 32 justice juvenile department delinquency thursday descriptor council p.m. service human in

○ FR881109-0149
○ FR: Coordinating Council; Meeting
○ FR88602-0134
○ FR: Coordinating Council on Juvenile Justice and Delinquency Prevention; Meeting
○ FR88609-0103
○ FR88609-0031

Figure 5: The interface in cluster mode.

A Appendix: Interactive Track System Description

A.1 System Description

The Interactive Track system is described in the body of this report.

A.2 Experimental Conditions

The experimental conditions are described to a large extent in the body of this report. The required information is repeated here.

1. Searcher characteristics

   a. Number of searchers in experiment:

      4

   b. Number of searchers per topic:

      2

   c. Age/age group of searchers:

      20 – 45

   d. IR searching experience of searchers.

      Familiarity with online bibliographic catalogs.

   e. Educational level of searchers.

      Bachelors and Masters degrees

   f. Undergraduate major of searchers.

      Not known.

   g. Experience/familiarity with subject of topic.

      Not known.

   h. Work affiliation of searchers.

      UC Berkeley graduate students.

2. Task description

   Essentially, the description suggested by the leaders of the TREC interactive track: *Find as many good documents as you can for a topic, in around 30 minutes, without collecting too much rubbish.*

3. Training

   Participants completed queries in two sessions. The experiments were run in an otherwise empty room with a video camera recording the session. Participants were given an 10-minute demonstration of the interface followed by a 10-minute warmup exersize. They were also provided with a 3 page description of the interface for reference. A binder of topic descriptions was prepared for each participant, with each topic description appeared on the top of a separate page. Participants were not allowed to look at a new topic before the current one was completed.

III. Search process
Note: the numbers below are only for XERINT2.

1. Elapsed time in seconds per search

   Mean: 1563

   Median: 1627

   SD: 293

   Range: 1104 – 1913

2. Number of documents "viewed" in during the search.

   A document is considered viewed if the user explicitly views the document's contents, as opposed to just seeing the title (and optionally, the associated TileBar).

   Mean: 17.6

   Median: 12

   SD: 14.4

   Range: 4 – 61

3. Number of iterations per search.

   A new iteration takes place when a new search is run for a particular query.

   Mean: 2.1

   Median: 2

   SD: 1.5

   Range: 1 – 6

4. Number of terms used in queries.

   Mean: 5.13

   Median: 8

   SD: 9.3

   Range: 4 – 38

5. Use of system features.

   N/A.

6. Number of user errors made per search.

   N/A.

7. Search narrative for topic 236 (and 216).

   See Appendix B.

B Appendix: Transcripts for Selected Topics

The first number shown on almost every line is the time in seconds. Next appears the mode the user was in, one of TITLES, TILEBARS, or CLUSTERS (for Scatter/Gather). In some cases lists of documents appear in the order in which they were ranked, left to right and top to bottom. Only the top few documents' are shown in each mode (except after the initial search, when all are shown). A 1 indicates a document judged relevant and a 0 a document judged irrelevant. When the participant views or selects a document, it's relevance judgment is also shown.

B.1 Transcript for Topic 216

The first example transcript is that of Topic 216, the running example of this paper.

```
22 TITLES     1 Changing mode to TITLES
80 NEWSEARCH   2
Running new search.
Num docs: 2S0
Termsets: (OSTEOPOROSIS TREATMENT PREVENTION
           RESEARCH)
0 0 0 0 1 0 0 1 0 1 0 1 0 0 1 1 1 0 1 0 0 0 1 0 0
0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 1 0 0 0 0 0 0 0
0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0


 181 TITLES     3 Changing mode to TITLES
 206 TITLES     3 Visible Contents of Titles
  ((TITLES) (41)
   (0 0 0 0 1 0 0 1 0 1 0 1 0 0 1 1 1 0 1 0 0
    0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1))

 210 CLUSTERS  4 Changing mode to CLUSTERS
 270 CLUSTERS  4 Visible Contents of Cluster 0
   (DIAGNOSIS FAILURE DESCRIPTOR WASTE ROOF
    MANAGEMENT ANALYSIS DIALYSIS WATER
   CONVENTIONAL DATA U.S.)
   (0 0 0 0 0 0 0 0 0 0 0)

 274 CLUSTERS  4 Visible Contents of Cluster 1
   (OFFICE DEPARTMENT JUSTICE JUVENILE CHILD EDUCATION
    DELINQUENCY WASHINGTON BILL SECRETARY AGENCY GRANT)
   (0 0 0 0 0 0 0 0 0 0 0)

 278 CLUSTERS  4 Visible Contents of Cluster 2
   (INSTITUTE COMMITTEE ADMINISTRATION MENTAL A.M.
    BUILD BILLION ROCKVILLE TIME DEVELOPMENT ROOM
    ALCOHOL)
   (0 0 0 0 0 0 0 0 0 0 0)

 280 CLUSTERS  4 Visible Contents of Cluster 3
   (BONE OSTEOPOROSIS CALCIUM AGE LOSS PREVENT
   FRACTURE UNIVERSITY MENOPAUSE ESTROGEN DR. HORMONE)
   (1 1 1 1 1 1 1 1 1 0 1)

 319 CLUSTERS  4 Selecting Cluster 3
 323 TILEBARS  5 Changing mode to TILEBARS
 332 TILEBARS  5 Visible Contents of Tilebars
   (0 0 0 1 1 1 1 1 1 0 0 1 0 1 1 1 1 1)
```

```
 341 TILEBARS  S Showing doc FR88S13-01S7 Tile 7
 383 TILEBARS  S Showing doc FR88S13-01S7 Tile 9
 421 TILEBARS  5 Selecting doc FR88S13-0157 0
 444 TILEBARS  S Showing doc AP881110-0214 Tile 0
 468 TILEBARS  5 Showing doc AP881128-0187 Tile 0
 S05 TILEBARS  S Selecting doc AP881128-0187 0
 514 TILEBARS  S Showing doc WSJ900S03-0011 Tile 0
 514 TILEBARS  S Showing doc WSJ900S03-0011 Tile 0
 S27 TILEBARS  S Selecting doc WSJ900S03-0011 1
 S30 TILEBARS  S Selecting doc AP900712-0031 1
 S39 TILEBARS  5 Showing doc SJMN91-063S3137 Tile 0
 S40 TILEBARS  5 Showing doc SJMN91-063S3137 Tile 0
 S7S TILEBARS  S Selecting doc SJMN91-0629S004 1
 580 TILEBARS  5 Showing doc WSJ900712-0096 Tile 0
 S80 TILEBARS  S Showing doc WSJ900712-0096 Tile 0
 592 TILEBARS  5 Selecting doc WSJ900712-0096 1
 S97 TILEBARS  S Showing doc AP900319-0222 Tile 0
 598 TILEBARS  5 Showing doc AP900319-0222 Tile 0
 620 TILEBARS  5 Selecting doc AP900319-0222 0
 638 TILEBARS  S Showing doc ZF32-1S0-197 Tile 0
 638 TILEBARS  S Showing doc ZF32-1S0-197 Tile 0
 668 TILEBARS  5 Selecting doc ZF32-1S0-197 0
 670 TILEBARS  5 Showing doc AP900S02-0083 Tile 0
 670 TILEBARS  S Showing doc AP900502-0083 Tile 0
 671 TILEBARS  S Showing doc AP900S02-0083 Tile 0
 677 TILEBARS  S Selecting doc AP900S02-0083 1
 682 TILEBARS  S Showing doc AP900S17-0238 Tile 0
 696 TILEBARS  S Selecting doc AP900S17-0238 0
 699 TILEBARS  S Selecting doc AP881202-0027 1
 703 TILEBARS  S Selecting doc AP900927-0033 1
 70S TILEBARS  S Showing doc WSJ911031-001S Tile 0
 727 TILEBARS  S Selecting doc WSJ911031-001S 1
 731 TILEBARS  5 Selecting doc AP900503-001S 1
 73S TILEBARS  5 Selecting doc SJMN91-062751S7 1
 738 TILEBARS  5 Showing doc AP881212-0266 Tile 0
 748 TILEBARS  S Selecting doc AP881212-0266 1
 7S1 TILEBARS  5 Showing doc SJMN91-06326216 Tile 0
 7S1 TILEBARS  S Showing doc SJMN91-06326216 Tile 0
 783 TILEBARS  5 Showing doc AP880603-0121 Tile 0
 783 TILEBARS  5 Showing doc AP880603-0121 Tile 0
 848 TILEBARS  S Showing doc AP900616-0022 Tile 0
 849 TILEBARS  S Showing doc AP900616-0022 Tile 0
 8S4 TILEBARS  S Showing doc AP900616-0022 Tile 1
 86S TILEBARS  S Selecting doc AP900616-0022 0
 869 TILEBARS  S Showing doc SJMN91-06111119 Tile 0
 898 TILEBARS  S Selecting doc AP90040S-0179 1
 904 TILEBARS  S Showing doc SJMN91-06340009 Tile 0
 90S TILEBARS  5 Showing doc SJMN91-06340009 Tile 0
 929 TILEBARS  S Selecting doc SJMN91-06340009 1
 934 TILEBARS  S Showing doc SJMN91-06262031 Tile 0
 935 TILEBARS  5 Showing doc SJMN91-06262031 Tile 0
 946 TILEBARS  5 Selecting doc SJMN91-06262031 1
 948 TILEBARS  5 Unselecting doc SJMN91-06262031 1
 952 TILEBARS  S Showing doc ZF32-3S3-056 Tile 0
 963 TILEBARS  5 Selecting doc ZF32-3S3-056 0
 966 TILEBARS  S Showing doc SJMN91-06074018 Tile 0
 966 TILEBARS  S Showing doc SJMN91-06074018 Tile 0
 978 TILEBARS  5 Showing doc AP881212-0255 Tile 0
 979 TILEBARS  5 Showing doc AP881212-0255 Tile 0
1027 TILEBARS  5 Backup up state
1028 CLUSTERS  4 Changing mode to CLUSTERS
1037 CLUSTERS  4 Visible Contents of Cluster 0
   (DIAGNOSIS FAILURE DESCRIPTOR WASTE ROOF MANAGEMENT
    ANALYSIS DIALYSIS WATER CONVENTIONAL DATA U.S.)
   (0 0 0 0 0 0 0 0 0 0 0)

1041 CLUSTERS  4 Visible Contents of Cluster 1
   (OFFICE DEPARTMENT JUSTICE JUVENILE CHILD EDUCATION
    DELINQUENCY WASHINGTON BILL SECRETARY AGENCY GRANT)
   (0 0 0 0 0 0 0 0 0 0 0)

1045 CLUSTERS  4 Visible Contents of Cluster 2
   (INSTITUTE COMMITTEE ADMINISTRATION MENTAL A.M.
   BUILD BILLION ROCKVILLE TIME DEVELOPMENT ROOM ALCOHOL)
   (0 0 0 0 0 0 0 0 0 0 0)

1047 CLUSTERS  4 Visible Contents of Cluster 3
   (BONE OSTEOPOROSIS CALCIUM AGE LOSS PREVENT FRACTURE
    UNIVERSITY MENOPAUSE ESTROGEN DR. HORMONE)
   (1 1 1 1 1 1 1 1 1 0 1)

1058 CLUSTERS  4 Selecting Cluster 2
1061 TILEBARS  6 Changing mode to TILEBARS
```

```
1072 TILEBARS  6 Visible Contents of Tilebars
   (0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)

1077 TILEBARS  6 Showing doc SJMN91-06059075 Tile 0
1118 TITLES    7 Changing mode to TITLES
1132 TITLES    7 Visible Contents of Titles
  ((TITLES) (41)
   (0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
    0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0))

1247 NEWSEARCH  8
Running new search.
Num docs: 500
Termsets: (OSTEOPOROSIS TREATMENT PREVENTION
          RESEARCH)
0 0 0 0 1 0 0 1 0 1 0 1 0 0 1 1 1 0 1 0 0 0 1 0 0
0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 1 0 0 0 0 0 0 0
0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0


1247 TITLES    9 Changing mode to TITLES
1309 TITLES    9 Visible Contents of Titles
  ((TITLES) (41)
   (0 0 0 0 1 0 0 1 0 1 0 1 0 0 1 1 1 0 1 0 0 0 1 0
    0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1))

1316 CLUSTERS 10 Changing mode to CLUSTERS
1382 CLUSTERS 10 Visible Contents of Cluster 0
   (COMMITTEE APPLICATION OFFICE DEPARTMENT PUBLIC
    GRANT BUILD A.M. ROOM MENTAL ADVISORY CHILD)
   (0 0 0 0 0 0 0 0 0 0 0)

1387 CLUSTERS 10 Visible Contents of Cluster 1
   (STUDY WOMAN CANCER RESEARCHER BONE DR.
    UNIVERSITY AGE AMERICAN TEST OSTEOPOROSIS
    MEDICINE)
   (0 1 0 1 0 0 1 0 1 0 0)

1394 CLUSTERS 10 Visible Contents of Cluster 2
   (COMPANY TECHNOLOGY STREET PRODUCT WALL MARKET
    PRESIDENT PAGE GOVERNMENT BIOTECHNOLOGY
   SHARE DIRECTOR)
   (0 0 0 1 0 0 0 0 0 0 0)

1398 CLUSTERS 10 Visible Contents of Cluster 3
   (DESCRIPTOR DATA MANAGEMENT SCHEDULE USER CASE FAILURE
    GENERIC PROCESS INC. HEAT ORDER)
   (0 0 0 0 0 0 0 0 0 0 0)

1407 CLUSTERS 10 Selecting Cluster 1
1410 CLUSTERS 11 Changing mode to CLUSTERS
1425 CLUSTERS 11 Visible Contents of Cluster 0
   (BONE CALCIUM LOSS FRACTURE MENOPAUSE EXERCISE
    ESTROGEN MASS SUPPLEMENT DENSITY DIET INTAKE)
   (1 1 1 1 1 1 1 1 0 1 1)

1427 CLUSTERS 11 Visible Contents of Cluster 1
   (BREAST HEART FUND BILL NIH MONEY COMMITTEE
   BILLION WASHINGTON DIE ANIMAL CURE)
   (0 0 0 0 0 0 0 0 0 0 0)

1430 CLUSTERS 11 Visible Contents of Cluster 2
   (PATIENT PROGRAM VIRUS INFECTION CASE CLINICAL
    IMMUNE PHYSICIAN HOSPITAL INFORMATION
    NUMB GOVERNMENT)
   (0 0 0 0 0 0 0 0 0 0 0)
```

```
1431 CLUSTERS 11 Visible Contents of Cluster 3
   (CHILD GROWTH MUSCLE PATIENT DISORDER PILL
   MEDICATION LAW PARENT DYSTROPHY DIAGNOSIS
   INJECTION)
   (1 0 0 0 0 0 0 0 0 0)

1442 CLUSTERS 11 Selecting Cluster 0
1444 TILEBARS 12 Changing mode to TILEBARS
1455 TILEBARS 12 Visible Contents of Tilebars
   (0 0 1 1 1 1 1 1 1 0 1 1 1 1 1 0 0 0)

1467 TILEBARS 12 Showing doc WSJ900402-0142 Tile 0
1473 TILEBARS 12 Showing doc WSJ900402-0142 Tile 3
1490 TILEBARS 12 Showing doc ZF207-107-044 Tile 0
1502 TILEBARS 12 Showing doc SJMN91-06136178 Tile 0
1510 TILEBARS 12 Selecting doc SJMN91-06136178 1
1522 TILEBARS 12 Backup up state
1522 CLUSTERS 11 Changing mode to CLUSTERS
1526 CLUSTERS 11 Visible Contents of Cluster 0
   (BONE CALCIUM LOSS FRACTURE MENOPAUSE EXERCISE
    ESTROGEN MASS SUPPLEMENT DENSITY DIET INTAKE)
   (1 1 1 1 1 1 1 1 0 1 1)

1528 CLUSTERS 11 Visible Contents of Cluster 1
   (BREAST HEART FUND BILL NIH MONEY COMMITTEE BILLION
    WASHINGTON DIE ANIMAL CURE)
   (0 0 0 0 0 0 0 0 0 0 0)

1534 CLUSTERS 11 Visible Contents of Cluster 2
   (PATIENT PROGRAM VIRUS INFECTION CASE CLINICAL
    IMMUNE  PHYSICIAN HOSPITAL INFORMATION
    NUMB GOVERNMENT)
   (0 0 0 0 0 0 0 0 0 0 0)

1535 CLUSTERS 11 Visible Contents of Cluster 3
   (CHILD GROWTH MUSCLE PATIENT DISORDER PILL
   MEDICATION LAW PARENT DYSTROPHY DIAGNOSIS
   INJECTION)
   (1 0 0 0 0 0 0 0 0 0)

1536 CLUSTERS 11 Selecting Cluster 1
1540 TILEBARS 13 Changing mode to TILEBARS
1549 TILEBARS 13 Visible Contents of Tilebars
   (0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1)

1559 TILEBARS 13 Showing doc AP901106-0019 Tile 2
1572 TILEBARS 13 Selecting doc AP901106-0019 1
1576 TILEBARS 13 Showing doc AP900502-0035 Tile 1
1603 TILEBARS 13 Showing doc WSJ910403-0155 Tile 5
1610 TILEBARS 13 Showing doc WSJ910403-0155 Tile 5
1627 TILEBARS 13 Selecting doc WSJ910403-0155 1
1631 TILEBARS 13 Showing doc SJMN91-06175092 Tile 0
1651 TILEBARS 13 Selecting doc SJMN91-06175092 1
1664 TILEBARS 13 Showing doc SJMN91-06326216 Tile 2
1684 TILEBARS 13 Showing doc WSJ910619-0089 Tile 1
1979 TILEBARS 13 Query done

Final selected documents:

(FR88513-0157 0) (AP881128-0187 0)
(WSJ900503-0011 1) (AP900712-0031 1)
(SJMN91-06295004 1) (WSJ900712-0096 1)
 (AP900319-0222 0) (ZF32-150-197 0)
(AP900502-0083 1) (AP900517-0238 0)
(AP881202-0027 1) (AP900927-0033 1)
(WSJ911031-0015 1) (AP900503-0015 1)
(SJMN91-06275157 1) (AP881212-0266 1)
(AP900616-0022 0) (AP900405-0179 1)
(SJMN91-06340009 1) (ZF32-353-056 0)
(SJMN91-06136178 1) (AP901106-0019 1)
(WSJ910403-0155 1) (SJMN91-06175092 1)
```

B.2 Transcript for Topic 236

As with most of the Interactive Track systems at TREC4, the results on Topic 236 were not the strongest. In this case, only one search was run. Of the 8 possible relevant documents from the search, 6 of these were found and saved.

```
23 TITLES     1 Changing mode to TITLES
311 NEWSEARCH  2
Running new search.
Num docs: 500
Termsets: ({LAW LEGAL REGULATION} {SEA OCEAN} COMPARE
           {AGREEMENT DISAGREEMENT})
0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0


 314 TITLES     3 Changing mode to TITLES
 452 TITLES     3 Visible Contents of Titles
  ((TITLES) (41)
   (0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0
    0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0))

 463 CLUSTERS  4 Changing mode to CLUSTERS
 578 CLUSTERS  4 Visible Contents of Cluster 0
   (WATER AP DUMP SOVIET MARINE MILE COAST OIL NATIONAL
    WORLD VESSEL SITE)
  (0 1 0 0 0 0 0 0 1 0 0)

 587 CLUSTERS  4 Visible Contents of Cluster 1
   (FUND CC AVAILABILITY SCHEDULE CHECK APPLY DEPOSIT
    CHILD ACCOUNT AUTHORITY INSTITUTION LEA)
   (0 0 0 0 0 0 0 0 0 0 0)

 753 CLUSTERS  4 Visible Contents of Cluster 2
   (LEGAL COURT FIRM MR. LAWYER CASE JUSTICE JUDGE WALL
    SECURITY GOVERNMENT ATTORNEY)
   (0 0 0 0 0 0 0 0 0 0 0)

 773 CLUSTERS  4 Visible Contents of Cluster 3
   (COMMISSION PARTY MARITIME TITLE LINE INTEREST
   SYNOPSIS PORT DC FREIGHT LICENSE CONFERENCE)
   (0 0 0 0 0 0 0 0 0 0 0)

 813 CLUSTERS  4 Selecting Cluster 3
 817 CLUSTERS  4 Selecting Cluster 0
 823 CLUSTERS  5 Changing mode to CLUSTERS
 846 CLUSTERS  5 Visible Contents of Cluster 0
   (FOOT RESCUE GUARD SINK ABOARD CREW GAS SAFETY AIR
    CATTLE SPILL SOUTH)(0 0 0 0 0 0 0 0 0 0 0)

 856 CLUSTERS  5 Visible Contents of Cluster 1
   (SOVIET FISH WORLD TIME FISHERY SCIENTIST COUNTRY
    BERING UNION LION PACIFIC NATION)
   (1 1 0 0 0 0 0 1 0 0 0)

 877 CLUSTERS  5 Visible Contents of Cluster 2
   (COMMISSION PARTY MARITIME FILE TITLE LINE
    SECTION INTEREST SYNOPSIS CODE PORT SECRETARY)
   (0 0 0 0 0 0 0 0 0 0 0)
```

```
 887 CLUSTERS  5 Visible Contents of Cluster 3
   (DUMP SITE WASTE DISPOSAL DRILL SLUDGE EPA
    OFFER YORK ENVIRONMENTAL BAN COMPANY)
   (0 0 0 0 0 0 0 0 0 0 0)

 889 CLUSTERS  5 Showing doc FR88512-0056 Tile 0
 956 CLUSTERS  5 Showing doc FR881018-0063 Tile 0
 989 CLUSTERS  5 Showing doc FR88607-0070 Tile 0
1017 CLUSTERS  5 Selecting Cluster 0
1020 CLUSTERS  5 Selecting Cluster 3
1025 CLUSTERS  5 Selecting Cluster 1
1030 CLUSTERS  6 Changing mode to CLUSTERS
1051 CLUSTERS  6 Visible Contents of Cluster 0
   (CONTAINER GAS TEMPLE DEAD FERRY PETROLEUM
   SCROLL BID RIG GOVERNMENT ASSET EXPLOSION)
   (0 0 0 0 0 0 0 0 0 0 0)

1065 CLUSTERS  6 Visible Contents of Cluster 1
   (U.S. FISH SOVIET SHIP UNITE OFFICIAL VESSEL
    WORLD TIME WASHINGTON SCIENTIST ISLAND)
   (1 0 0 0 1 0 0 1 0 0 0)

1067 CLUSTERS  6 Visible Contents of Cluster 2
   (WASTE SLUDGE BAN INDUSTRIAL BILL POLLUTION
   JERSEY MEDICAL SEWAGE HOUSE SENATE CONGRESS)
   (0 0 0 0 0 0 0 0 0 0 0)

1071 CLUSTERS  6 Visible Contents of Cluster 3
   (DRILL MURPHY MANAGEMENT DREDGE MATERIAL IMPACT
    LA-5 DEEP ADMINISTRATION ACTION
    DESIGNATION PROGRAM)
   (0 0 0 0 0 0 0 0 0 0 0)

1074 CLUSTERS  6 Selecting doc AP880810-0056 0
1076 CLUSTERS  6 Selecting doc AP880809-0150 0
1080 CLUSTERS  6 Selecting doc AP881118-0130 0
1084 CLUSTERS  6 Selecting doc AP881018-0197 0
1086 CLUSTERS  6 Selecting doc AP881008-0018 0
1089 CLUSTERS  6 Selecting doc AP901029-0103 0
1101 CLUSTERS  6 Selecting doc AP901103-0030 0
1102 CLUSTERS  6 Unselecting doc AP901103-0030 0
1102 CLUSTERS  6 Selecting doc AP901103-0030 0
1113 CLUSTERS  6 Selecting doc FR881129-0025 0
1116 CLUSTERS  6 Selecting doc SJMN91-06011130 0
1120 CLUSTERS  6 Selecting doc AP881213-0090 0
1133 CLUSTERS  6 Selecting doc AP880830-0087 0
1140 CLUSTERS  6 Selecting doc SJMN91-06254147 0
1141 CLUSTERS  6 Selecting doc AP900404-0116 0
1158 CLUSTERS  6 Selecting doc WSJ910618-0147 0
1159 CLUSTERS  6 Selecting doc WSJ910124-0116 0
1197 CLUSTERS  6 Selecting Cluster 3
1201 CLUSTERS  6 Selecting Cluster 1
1209 CLUSTERS  6 Selecting Cluster 0
1213 CLUSTERS  7 Changing mode to CLUSTERS
1226 CLUSTERS  7 Visible Contents of Cluster 0
   (OIL DRILL EXPLORATION GAS MURPHY PETROLEUM
   ENERGY JOURNAL WALL STREET PET PAGE)
   (0 0 0 0 0 0 0 0 0 0 0)

1233 CLUSTERS  7 Visible Contents of Cluster 1
   (SITE DISPOSAL MANAGEMENT DEEP IMPACT DREDGE
    MATERIAL LAVA PROGRAM LA-5 HOME ADMINISTRATION)
   (0 0 0 0 0 0 0 0 0 0 0)

1249 CLUSTERS  7 Visible Contents of Cluster 2
   (U.S. SOVIET SHIP UNITE VESSEL OFFICIAL WASHINGTON
    ISLAND FISHERY INTERNATIONAL COUNTRY BERING)
   (1 1 0 0 0 0 0 1 0 0 0)

1256 CLUSTERS  7 Visible Contents of Cluster 3
   (CONTAINER TEMPLE SCROLL FERRY SALE DEAD ASSET
    STENA BILLION BUY TIPHOOK RECAPITALIZATION)
   (0 0 0 0 0 0 0 0 0 0 0)

1257 CLUSTERS  7 Selecting Cluster 0
1259 TILEBARS  8 Changing mode to TILEBARS
1271 TILEBARS  8 Visible Contents of Tilebars
   (0 0 0 0 0 0 0 0 0 0 0 0 0)

1278 TILEBARS  8 Selecting doc SJMN91-06108037 0
1284 TILEBARS  8 Unselecting doc WSJ910124-0116 0
1289 TILEBARS  8 Unselecting doc WSJ910618-0147 0
1290 TILEBARS  8 Selecting doc WSJ910618-0147 0
```

```
1290 TILEBARS  8 Unselecting doc WSJ910618-0147 0
1299 TILEBARS  8 Selecting doc AP880930-0025 0
1301 TILEBARS  8 Selecting doc AP880929-0298 0
1305 TILEBARS  8 Backup up state
1305 CLUSTERS  7 Changing mode to CLUSTERS
1310 CLUSTERS  7 Visible Contents of Cluster 0
   (OIL DRILL EXPLORATION GAS MURPHY PETROLEUM
   ENERGY JOURNAL WALL STREET PET PAGE)
   (0 0 0 0 0 0 0 0 0 0 0)

1314 CLUSTERS  7 Visible Contents of Cluster 1
   (SITE DISPOSAL MANAGEMENT DEEP IMPACT DREDGE
    MATERIAL LAVA PROGRAM LA-5 HOME ADMINISTRATION)
   (0 0 0 0 0 0 0 0 0 0 0)

1324 CLUSTERS  7 Visible Contents of Cluster 2
   (U.S. SOVIET SHIP UNITE VESSEL OFFICIAL WASHINGTON
   ISLAND FISHERY INTERNATIONAL COUNTRY BERING)
   (1 1 0 0 0 0 0 1 0 0 0)

1326 CLUSTERS  7 Visible Contents of Cluster 3
   (CONTAINER TEMPLE SCROLL FERRY SALE DEAD ASSET
   STENA BILLION BUY TIPHOOK RECAPITALIZATION)
   (0 0 0 0 0 0 0 0 0 0 0)

1327 CLUSTERS  7 Selecting Cluster 3
1330 TILEBARS  9 Changing mode to TILEBARS
1347 TILEBARS  9 Visible Contents of Tilebars
   (0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)

1358 TILEBARS  9 Backup up state
1359 CLUSTERS  7 Changing mode to CLUSTERS
1362 CLUSTERS  7 Visible Contents of Cluster 0
   (OIL DRILL EXPLORATION GAS MURPHY PETROLEUM
    ENERGY JOURNAL WALL STREET PET PAGE)
   (0 0 0 0 0 0 0 0 0 0 0)

1364 CLUSTERS  7 Visible Contents of Cluster 1
   (SITE DISPOSAL MANAGEMENT DEEP IMPACT DREDGE
   MATERIAL LAVA PROGRAM LA-5 HOME ADMINISTRATION)
   (0 0 0 0 0 0 0 0 0 0 0)

1371 CLUSTERS  7 Visible Contents of Cluster 2
   (U.S. SOVIET SHIP UNITE VESSEL OFFICIAL WASHINGTON
   ISLAND FISHERY INTERNATIONAL COUNTRY BERING)
   (1 1 0 0 0 0 0 1 0 0 0)

1372 CLUSTERS  7 Visible Contents of Cluster 3
   (CONTAINER TEMPLE SCROLL FERRY SALE DEAD ASSET
    STENA BILLION BUY TIPHOOK RECAPITALIZATION)
   (0 0 0 0 0 0 0 0 0 0 0)

1372 CLUSTERS  7 Selecting Cluster 1
1374 CLUSTERS  7 Selecting Cluster 2
1378 CLUSTERS 10 Changing mode to CLUSTERS
1392 CLUSTERS 10 Visible Contents of Cluster 0
   (SITE FOOT DISPOSAL CATTLE DEEP IMPACT PEN DREDGE
    MATERIAL SQUARE RULE LA-5)
   (0 0 0 0 0 0 0 0 0 0 0)

1394 CLUSTERS 10 Visible Contents of Cluster 1
   (SOVIET AGREEMENT BERING COUNTRY UNION INTERNATIONAL
   AMERICAN NATION SIGN FOREIGN MOSCOW ZONE)
   (1 1 0 0 0 0 1 0 0 0 0)

1401 CLUSTERS 10 Visible Contents of Cluster 2
   (LION MAMMAL OTTER CALIFORNIA BOAT ANIMAL
    HOUSE CITY SEAL FEDERAL KILL COUNTY)
   (0 0 0 0 0 0 0 0 0 0 0)

1406 CLUSTERS 10 Visible Contents of Cluster 3
   (RESCUE GUARD OIL CREW ABOARD SINK SOUTH
    HELICOPTER AIRCRAFT SEARCH FUEL CRUISE)
   (0 0 0 0 0 0 0 0 0 0 0)

1423 CLUSTERS 10 Showing doc AP900527-0035 Tile 0
1429 CLUSTERS 10 Selecting Cluster 0
1431 TILEBARS 11 Changing mode to TILEBARS
1438 TILEBARS 11 Visible Contents of Tilebars
   (0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)

1443 TILEBARS 11 Selecting doc FR88826-0028 0
1448 TILEBARS 11 Selecting doc FR881017-0004 0
```

```
1454 TILEBARS 11 Selecting doc SJMN91-06151045 0
1463 TILEBARS 11 Selecting doc SJMN91-06263151 0
1481 TILEBARS 11 Backup up state
1481 CLUSTERS 10 Changing mode to CLUSTERS
1484 CLUSTERS 10 Visible Contents of Cluster 0
   (SITE FOOT DISPOSAL CATTLE DEEP IMPACT PEN DREDGE
   MATERIAL SQUARE RULE LA-5)
  (0 0 0 0 0 0 0 0 0 0 0)

1492 CLUSTERS 10 Visible Contents of Cluster 1
   (SOVIET AGREEMENT BERING COUNTRY UNION INTERNATIONAL
    AMERICAN NATION SIGN FOREIGN MOSCOW ZONE)
   (1 1 0 0 0 0 1 0 0 0 0)

1493 CLUSTERS 10 Visible Contents of Cluster 2
   (LION MAMMAL OTTER CALIFORNIA BOAT ANIMAL
    HOUSE CITY SEAL FEDERAL KILL COUNTY)
   (0 0 0 0 0 0 0 0 0 0 0)

1494 CLUSTERS 10 Visible Contents of Cluster 3
   (RESCUE GUARD OIL CREW ABOARD SINK SOUTH
    HELICOPTER AIRCRAFT SEARCH FUEL CRUISE)
   (0 0 0 0 0 0 0 0 0 0 0)

1495 CLUSTERS 10 Selecting Cluster 1
1498 TILEBARS 12 Changing mode to TILEBARS
1511 TILEBARS 12 Visible Contents of Tilebars
   (0 1 0 0 1 0 0 1 1 0 0 0 1 0 0 0 0 0)

1513 TILEBARS 12 Selecting doc FR88616-0064 0
1517 TILEBARS 12 Selecting doc AP881101-0176 1
1523 TILEBARS 12 Selecting doc AP900601-0165 0
1528 TILEBARS 12 Selecting doc AP900212-0157 0
1538 TILEBARS 12 Selecting doc AP881111-0203 0
1542 TILEBARS 12 Selecting doc AP880419-0033 1
1545 TILEBARS 12 Selecting doc AP900831-0147 0
1552 TILEBARS 12 Selecting doc AP881025-0139 1
1556 TILEBARS 12 Selecting doc AP880518-0301 1
1563 TILEBARS 12 Selecting doc AP880817-0026 0
1579 TILEBARS 12 Selecting doc AP880423-0166 0
1582 TILEBARS 12 Selecting doc AP881024-0151 0
1586 TILEBARS 12 Selecting doc AP880425-0174 0
1589 TILEBARS 12 Selecting doc AP900129-0102 0
1596 TILEBARS 12 Unselecting doc AP900129-0102 0
1599 TILEBARS 12 Selecting doc AP881228-0105 1
1605 TILEBARS 12 Selecting doc AP900402-0131 0
1612 TILEBARS 12 Selecting doc AP900109-0043 1
1640 TILEBARS 12 Backup up state
1640 CLUSTERS 10 Changing mode to CLUSTERS
1647 CLUSTERS 10 Visible Contents of Cluster 0
   (SITE FOOT DISPOSAL CATTLE DEEP IMPACT PEN
    DREDGE MATERIAL SQUARE RULE LA-5)
   (0 0 0 0 0 0 0 0 0 0 0)

1653 CLUSTERS 10 Visible Contents of Cluster 1
   (SOVIET AGREEMENT BERING COUNTRY UNION
    INTERNATIONAL AMERICAN NATION SIGN FOREIGN
    MOSCOW ZONE)
   (1 1 0 0 0 0 1 0 0 0 0)

1655 CLUSTERS 10 Visible Contents of Cluster 2
   (LION MAMMAL OTTER CALIFORNIA BOAT ANIMAL
    HOUSE CITY SEAL FEDERAL KILL COUNTY)
   (0 0 0 0 0 0 0 0 0 0 0)

1656 CLUSTERS 10 Visible Contents of Cluster 3
   (RESCUE GUARD OIL CREW ABOARD SINK SOUTH
    HELICOPTER AIRCRAFT SEARCH FUEL CRUISE)
   (0 0 0 0 0 0 0 0 0 0 0)

1658 CLUSTERS 10 Selecting Cluster 2
1665 CLUSTERS 10 Selecting Cluster 3
1668 TILEBARS 13 Changing mode to TILEBARS
1684 TILEBARS 13 Visible Contents of Tilebars
   (0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)

1686 TILEBARS 13 Selecting doc FR88927-0030 0
1701 TILEBARS 13 Selecting doc AP900517-0140 0
1730 TILEBARS 13 Selecting doc SJMN91-06212079 0
1732 TILEBARS 13 Selecting doc SJMN91-06219052 0
1742 TILEBARS 13 Selecting doc AP900419-0113 0
1867 TILEBARS 13 Query done
```

```
Final selected documents:

(AP880810-0056 0) (AP880809-0150 0)
(AP881118-0130 0) (AP881018-0197 0)
(AP881008-0018 0) (AP901029-0103 0)
(AP901103-0030 C) (FR881129-0025 0)
(SJMN91-06011130 0) (AP881213-0090 0)
(AP880830-0087 0) (SJMN91-06254147 0)
(AP900404-0116 0) (SJMN91-06108037 0)
(AP880930-0025 0) (AP880929-0298 0)
(FR88826-0028 0) (FR881017-0004 0)
(SJMN91-06151045 0) (SJMN91-06263151 0)
(FR88616-0064 0) (AP881101-0176 1)
(AP900601-0165 0) (AP900212-0157 0)
(AP881111-0203 0) (AP880419-0033 1)
(AP900831-0147 0) (AP881025-0139 1)
(AP880518-0301 1) (AP880817-0026 0)
(AP880423-0166 0) (AP881024-0151 0)
(AP880425-0174 0) (AP881228-0105 1)
(AP900402-0131 0) (AP900109-0043 1)
(FR88927-0030 0) (AP900517-0140 0)
(SJMN91-06212079 0) (SJMN91-06219052 0)
(AP900419-0113 0)
```

Siemens TREC-4 Report: Further Experiments with Database Merging

Ellen M. Voorhees
Siemens Corporate Research, Inc.
Princeton, NJ
ellen@scr.siemens.com

Abstract

A database merging technique is a strategy for combining the results of multiple, independent searches into a single cohesive response. An isolated database merging technique selects the number of documents to be retrieved from each database without using data from the component databases at run-time. In this paper we investigate the effectiveness of two isolated database merging techniques in the context of the TREC-4 database merging task. The results show that on average a merged result contains about 1 fewer relevant document per query than a comparable single collection run when retrieving up to 100 documents.

1 Introduction

Siemens has used TREC-4 to continue its investigation of the *collection fusion* or *database merging* problem. Informally, the database merging problem is to combine the retrieval results from multiple, independent databases into a single result that has the best possible effectiveness. Such a search is necessary in a variety of distributed IR settings, with the setting determining the kinds of data available to the merging strategies. We assume the merging process is dispatched by an entity that has no control over the individual databases. Therefore, we assume the only information the merging algorithm can obtain from a collection is a ranked list of documents in response to a query. We call merging strategies that have no other access to the individual databases *isolated* merging strategies. In contrast, the methods explored by Callan, Lu, and Croft [3] assume access to particular data items (e.g., word frequencies) within the individual databases. We call these strategies *integrated* merging strategies.

Since integrated strategies have access to more information, they can be expected to be more effective than isolated strategies. While in principle even isolated strategies can produce merged results that are more effective than the result obtained when searching the entire set of documents as a single collection [6], in practice the merged results produced by isolated strategies have been less effective than the single collection run. Our main goal is to minimize this degradation in isolated merging strategies.

TREC-4 contains a database merging track. The track defined the set of component databases to be searched, and stipulated that a single collection run be made to serve as a directly comparable baseline. Our single collection run, `siems1`, is also our ad hoc submission. Siemens runs `siems2` and `siems3` are database merging runs created using two different isolated merging strategies. Each of these runs is described in more detail below. Siemens did not perform any routing runs, nor did it participate in any other tracks.

The results of the experiments show a small degradation in the effectiveness of the merging runs as compared to the single collection run for moderate numbers of retrieved documents. Using the average of the precision after 20 documents are retrieved as the measure of effectiveness, the database merging runs are 10% and 13% less effective than the single collection run. That is, on average the merged runs find approximately 3/4 of a document fewer relevant documents in the top 20 retrieved documents per query than does the single collection run. After 100 documents retrieved, the percentage decreases are 12% and 17%. The average non-interpolated average precision over all relevant documents (with 1000 documents retrieved per query) degrades by 29% and 38%. Since users are generally interested in only a relatively small number of top-ranked documents, these techniques offer a viable solution to the database merging problem.

The next section describes Siemens's retrieval environment in general and the specific settings used for our TREC-4 runs. The following section provides a more detailed comparison between the effectiveness of the merged and single collection runs. Section 4 discusses the size of the data structures required to support the merging algorithms and the resulting efficiency of the merged search: since isolated strategies use no data from the individual collections to select the databases participating in the current search, and since the query is submitted only to those collections from which documents are desired, these merged searches are quite efficient. The final section explores some of the challenges created by the database merging task, and lists areas that still need to be addressed.

2 Merging and Retrieval Methods

Both of the database merging techniques used in this work use relevance assessments from past queries to select the number of documents to request from each database for the current query. If a non-zero number of documents, λ, is to be retrieved from a given database, the (natural language) query is submitted to that database and the λ most highly ranked documents are returned. The following datasets are thus required to test the effectiveness of these database merging methods.

A set of component databases. The database merging track chose the ten document sets contained on TREC disks two and three to be the set of databases to be searched. This choice was motivated by the fact that the union of the component databases is the set of documents to be used for the TREC-4 ad hoc task. The ten databases include: two AP newswire collections (1988 and 1990); a *Federal Register* collection; a set of U.S. patent disclosures; a *San Jose Mercury News* collection; three *Wall Street Journal* collections (1990, 1991, and 1992); and two collections of extracts from Ziff-Publishing's *Computer Selects* disks.

A set of training queries. We used TREC topics 1–200 as our training queries.

Training query retrieval results. Since the merging algorithms rely on ranked lists of documents, annotated with relevance data, to compute the number of documents to retrieve from a database, the training queries must be run against the individual databases and the retrieved relevant documents marked as such. The TREC collection does not contain relevance assessments for disk3 for topics 1–50 and 151–200, so some collections have more training queries than others. One of the objectives of this study is to investigate the performance of the merging strategies when the set of component databases has differing training data.

We use the SMART retrieval system from Cornell [1] as our underlying retrieval engine. In particular, we used the massive query expansion technique that produced good results for Cornell in TREC-3 [2]. The training query results were created by performing an initial run using 'lnc'-weighted document vectors and 'ltc'-weighted query vectors. The vectors were formed using the standard SMART indexing procedures, and they include both single terms and phrases. The top 15 retrieved documents for each query were *assumed* to be relevant (the actual relevance data was not used in this step), and were used to perform Rocchio feedback on the initial query. During the Rocchio feedback, the initial query was expanded with at most 100 new single terms and at most 10 new phrases; the Rocchio parameters were set to $\alpha = 8$, $\beta = 8$, and $\gamma = 0$. Once created, the newly expanded query was run against the collection to produce the retrieval results used in training.

A set of test queries. TREC topics 202–250 were the test topics.

Test query retrieval results. To form the actual merged result, the test queries must also be run against the individual databases. In this case, however, no relevance data is required (except for evaluation). The test query retrieval results were generated in the same way as the training query results except that a maximum of 500 single terms could be added to query. To conform with the TREC task requirements, 1000 documents were retrieved for each test query.

In addition to expanding the queries in each of the component database, queries 202–250 were expanded (with a maximum of 500 single terms) and run against the collection formed from the entire set of documents. This run is our ad hoc run, `siems1`, and provides a point of comparison for the merged runs.

Siemens run **`siems2`** was created using the Query Clustering (QC) merging technique, and run **`siems3`** was created using the Modeling Relevant Document Distributions (MRDD) merging technique. These are the same merging techniques used in our earlier work [4, 5, 6]. While the two techniques require the same basic datasets, described above, they use the training data in different ways. These differences are described in the following subsections.

2.1 Query Clustering

The basic steps in the query clustering merging method are presented in Figure 1. The training phase is depicted in Step 1 of the figure. For each database, the set of training queries that actually have relevance data in that database is clustered. The clusters are produced using Ward's algorithm and the inverse of the number of documents retrieved in common between two queries in the top 1000 documents as the distance metric. Query vectors are created in a vector space formed from the set of training queries, and the vectors of the queries contained within each cluster are averaged to create cluster centroids. (These are *unexpanded* query vectors — there are no documents to expand by.) Each cluster is also assigned a weight that reflects how effective queries in the cluster are on that database. The weight is computed as the average number of relevant documents retrieved by queries in the cluster, where a document is considered to be retrieved if it is in the top 100 documents.

Steps 2 and 3 of Figure 1 depict how the merged result is created for new queries. The cluster whose centroid vector is most similar to the query vector is selected for the query and the associated weight is returned. The set of weights returned by all the collections is used to apportion the retrieved set: when N documents are to be returned to the user and w_i is the weight returned by collection i, $\frac{w_i}{\sum_{i=1}^{C} w_i} * N$ documents are retrieved from collection i. The final ranking of the documents retrieved from each database is produced by a random process. To select the document for rank r, a collection is chosen by rolling a C-faced die that is biased by the number of documents

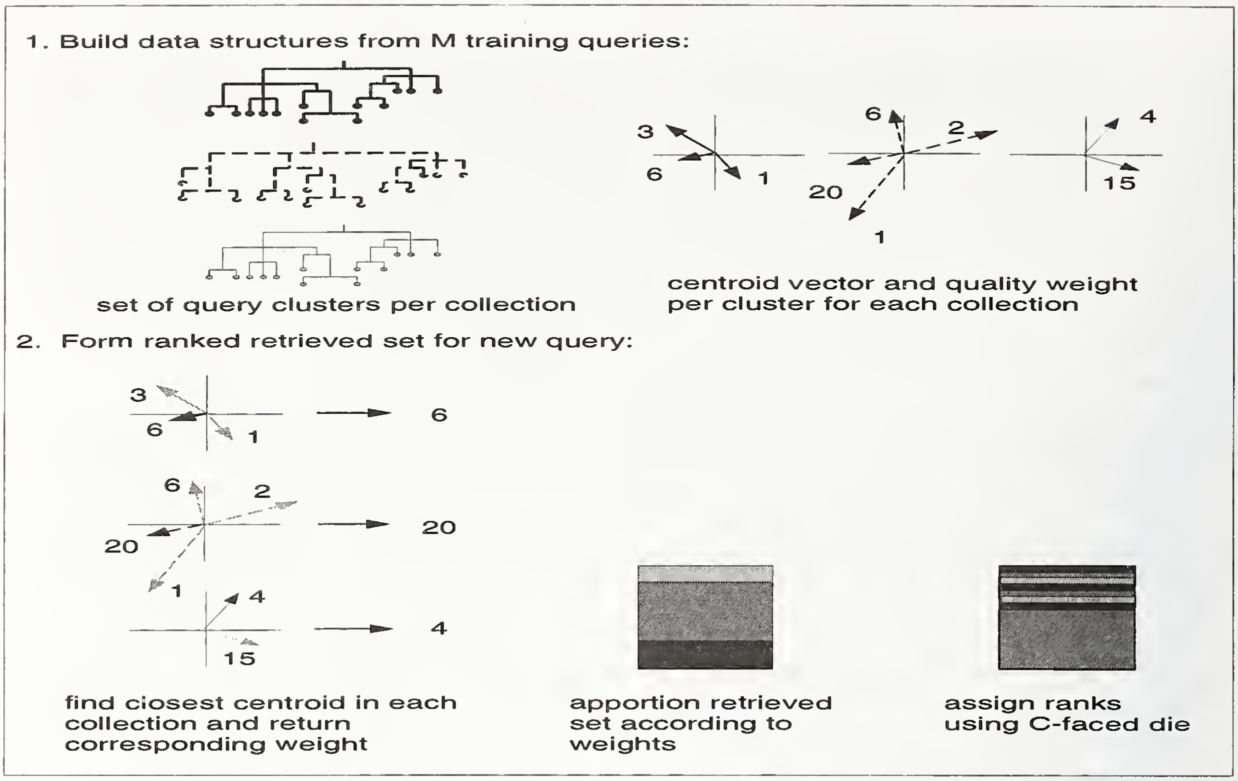

1. Build data structures from M training queries:
set of query clusters per collection
3
6
1
6
2
20
1
4
15
centroid vector and quality weight per cluster for each collection
2. Form ranked retrieved set for new query:
3
6
1
6
6
2
20
1
20
4
15
4
find ciosest centroid in each collection and return corresponding weight
apportion retrieved set according to weights
assign ranks using C-faced die

Figure 1: The QC database merging strategy.

still to be picked from each of the C collections. The next document from that collection is placed at rank r and removed from further consideration.

2.2 Modeling Relevant Document Distributions

Figure 2 summarizes the steps of the second database merging technique, modeling relevant document distributions (MRDD). Once again, the first step is a training phase. One set of (unexpanded) query vectors is created in a vector space constructed from all of the training queries. The system also stores an explicit representation of the relevant document distribution for each query in each database. This distribution is equivalent to the ranks of the relevant documents in the top 1000 retrieved.

The first step in processing a new query, q, is to determine the six training queries that are most similar to it. A model of q's retrieval behavior in each collection is constructed by averaging the relevant document distributions of these six nearest neighbors. However, since some databases do not have relevance assessments for all training queries, the average distribution in each database is actually computed over the set of nearest neighbors that have relevance data for that database, which may be less than six. Using the model distributions to predict the number of relevant documents that would be retrieved for q from each of the databases at different cut-off levels, the system computes the number of documents to retrieve from each collection such that the total number of relevant documents that would be retrieved is maximized. The "spill", the number of documents the maximization procedure computes will have no effect on the total number of relevant documents retrieved and may thus come from any collection, is distributed among the databases

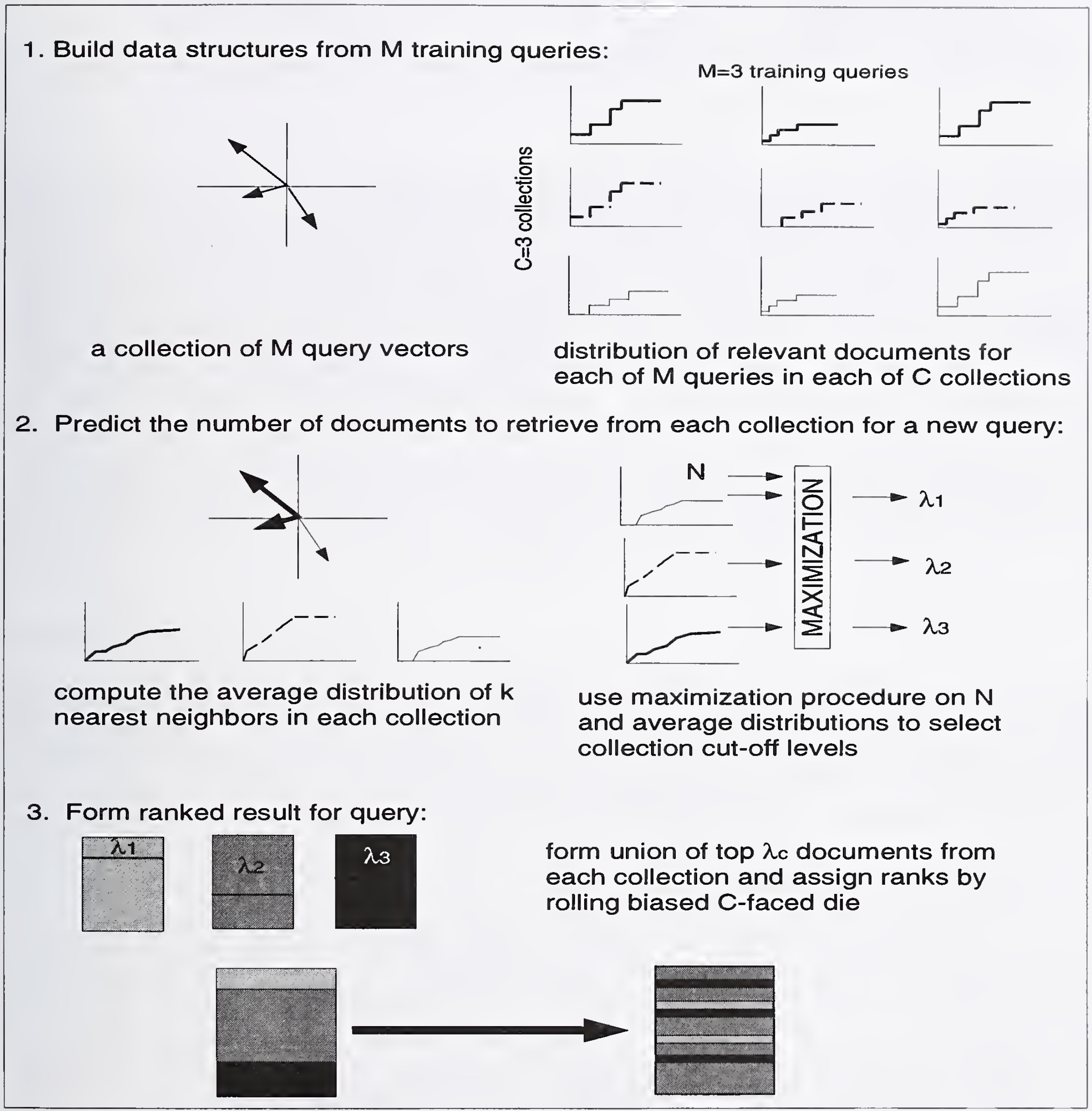

1. Build data structures from M training queries:
M=3 training queries
C=3 collections
a collection of M query vectors
distribution of relevant documents for each of M queries in each of C collections
2. Predict the number of documents to retrieve from each collection for a new query:
N
MAXIMIZATION
λ1
λ2
λ3
compute the average distribution of k nearest neighbors in each collection
use maximization procedure on N and average distributions to select collection cut-off levels
3. Form ranked result for query:
λ1
λ2
λ3
form union of top λc documents from each collection and assign ranks by rolling biased C-faced die

Figure 2: The MRDD database merging strategy.

	Ave. value	# best	# ≥ median	# worst
Relevant retrieved in top 100	24.65	0	30	1
Relevant retrieved in top 1000	77.3	7	38	0
Average precision	.2031	0	26	0

Table 1: Effectiveness of the ad hoc run siems1 as compared to other TREC-4 ad hoc runs.

in proportion to the number of documents that would be otherwise retrieved from it. The final ranking of the retrieved documents is produced by the same procedure as is used in the QC method.

The maximization procedure used above is an NP-complete optimization problem. In previous experiments with MRDD, a simple exhaustive search was efficient enough because the optimization entailed a small number of documents to be retrieved and/or a small number of databases. Unfortunately, with the TREC-4 submission deadline looming, we discovered that 1000 documents to be retrieved from a subset of 10 databases was prohibitively time-consuming to run. (And the same time pressures prevented an implementation of a more efficient optimization procedure.) Thus run siems3 was produced by telling the optimization procedure that 50 documents were to be retrieved and then multiplying the resulting number of documents to retrieve from each database by 20 to obtain a total of 1000 documents. Note that this is unlikely to seriously degrade the performance of the MRDD method. Previous experiments have shown that the model distributions are most accurate in the range of 20–100 documents to be retrieved [4, 5].

3 Retrieval Effectiveness

This section reports on the effectiveness of our retrieval runs. The single collection run siems1 meets the requirements of the TREC-4 ad hoc task (Topics 202–250 run against Disks 2 and 3), so the first subsection compares its effectiveness to the other ad hoc runs. The remainder of the section explores the effectiveness of the merged runs by comparing their effectiveness to that of siems1 and a variety of other baseline merging techniques.

3.1 Single Collection Results

Our ad hoc run siems1 is a completely automatic, single collection run that expanded queries as described above. The use of query expansion was motivated by our desire to have the searches of individual databases be as effective as possible, and to make comparisons meaningful we used the same technique for the single collection run. As the SMART group demonstrated in TREC-3 [2], query expansion improves retrieval performance by providing much more context in the queries. The context is created by adding the terms that occur in the documents retrieved in an initial phase to the newly expanded query vector.

The effectiveness of siems1 as compared to the other TREC-4 ad hoc runs is summarized in Table 1. The first column in the table gives the value obtained by siems1 averaged over the 49 ad hoc queries. The remaining columns give the number of queries for which siems1 obtained the best, the worst, and an above-median score.

In general, siems1 is an effective run, being at or above the median for a majority of the queries for all three effectiveness measures. (The one worst score was a query for which no relevant documents were retrieved when the median was one relevant document retrieved.) Unsurprisingly,

	Prec(20)		Prec(100)		Prec(1000)		Average Precision	
Single Collection (siems1)	.3265	—	.2465	—	.0773	—	.2031	—
QC Merging (siems2)	.2949	−10%	.2167	−12%	.0671	−13%	.1433	−29%
MRDD Merging (siems3)	.2847	−13%	.2053	−17%	.0536	−31%	.1253	−38%

Table 2: The effectiveness of the QC and MRDD merged results as compared to the single collection results.

the results demonstrate that the massive query expansion is a recall-oriented procedure: the queries tend to retrieve many relevant documents, but those documents are not always highly ranked. The siems1 run retrieved the most relevant documents in the top 1000 documents for seven queries, but the non-interpolated average precision was always much closer to the median.

Some queries were adversely affected by the automatic expansion procedure. This happened when short, non-relevant documents contained a key term of the query and were thus ranked highly in the initial set of retrieved documents. The automatic expansion based on these documents led the subsequent search in the wrong direction. For example, the retrieval performance of Query 248 in siems1 is much worse than the median performance. The text of Topic 248 is *What are some developments in electronic technology being applied to and resulting in advances for the blind.* Unfortunately, the *Wall Street Journal* has a number of very short earning reports for the company Electronic Technology. As a result, the final query had far more to do with finance than with blindness.

3.2 Database Merging Results

The single collection run provides one benchmark for the effectiveness of the database merging runs. As mentioned above, our goal is to have the effectiveness of the merged results match those of the single collection run. Table 2 gives the effectiveness of the single collection and merged runs averaged over the 49 queries. Effectiveness is measured in terms of the precision after 20, 100, and 1000 documents have been retrieved as well as the non-interpolated average precision. For the merged runs, the percentage difference over the single collection run is also given. As the non-interpolated average precision figures demonstrate, the merged runs are clearly less effective at ranking documents when large numbers of documents are retrieved than is the single collection run. The total number of relevant documents retrieved is much less severely degraded. Fortunately, the effectiveness is best at the smaller numbers of retrieved documents, which is the area most likely to be of concern to the typical user.

An important difference in these results is that the QC method is more effective than the MRDD method (the reverse was true in previous experiments). The MRDD method makes use of much more of the training data than the QC method: it stores and exploits the entire rankings of the training queries rather than summarizing their performance in a set of weights. Theoretically, this should lead to better performance for MRDD, and indeed that had been true [4]. However, such reliance on the training queries makes the method more susceptible to differences between the training and test queries. The topics in TREC-4 were much shorter than in previous years, and the subject matter of some topics did not always have corresponding training queries. In these cases, any test query words that just happened to be in training queries caused the resulting query-query similarity to be relatively large. For example, the text of Topic 224 is *What can be done to lower*

	Prec(20)		Prec(100)		Prec(1000)	
Single Collection (siems1)	.3265	—	.2465	—	.0773	—
Uniform	.2235	–32%	.1624	–34%	.0662	–14%
Optimal	.5663	+73%	.3478	+41%	n/a	—
AP Only	.3255	0%	.2173	–12%	.0458	–41%
QC Merging (siems2)	.2949	–10%	.2167	–12%	.0671	–13%
MRDD Merging (siems3)	.2847	–13%	.2053	–17%	.0536	–31%

Table 3: The effectiveness of the merged runs in comparison to a variety of benchmarks.

blood pressure for people diagnosed with high blood pressure? Include benefits and side effects. The most similar query to 224 matched only on *effect* and *includ*; the next most only on *diagnos*; and the third most similar on *side*, *high*, and *people*. As might be expected, none of these queries had anything to do with blood pressure. The query about diagnosis was vaguely medicine related, asking about computer programs that aided in medical diagnosis. As a result, MRDD retrieved 360 documents from the Ziff3 collection, which contains no relevant documents. The QC merging technique's more crude representation of topic areas makes it more robust against these types of errors.

There are several other benchmarks the merged runs can be compared against to obtain a fuller understanding of their effectiveness. Effectiveness measures for these baselines are given in Table 3. The *optimal* run uses relevance information to compute the best possible merged result given the retrieval results for the individual collections. As in previous experiments, the optimal merged run is significantly more effective than the single collection run. The *uniform* run retrieves an equal number of documents from each collection. This is essentially a straw-man benchmark. The uniform strategy is the best strategy to use in the absence of any training data; a viable merging strategy should be more effective than the uniform run. Since the uniform run is approximately as effective as the merged results after 1000 documents are retrieved, the behavior of the merging strategies at that large of number of retrieved documents is probably meaningless. The *AP-only* run retrieves half its documents from the AP88 collection and half from AP90. In previous experiments with the TREC collection, the queries exhibited a large bias towards the AP collection [6]. The results in Table 3 demonstrate that this bias exists for the TREC-4 queries as well.

This bias complicates the interpretation of the retrieval results. Learning to retrieve a majority of documents from the AP collection is a relatively simple thing to do, and the QC method learned to do just that. The QC method retrieved a majority of its documents from the AP collections for a sizable majority of the 49 queries. It also learned to completely ignore the patent and *Federal Register* collections. In these collections, the training queries clustered into a single large cluster that was assigned a weight of 0. New queries could therefore never retrieve documents from these collections. Such a strong bias against these collections is perfectly understandable given the relevance assessments for Topics 1–200.

4 Efficiency of Merging Techniques

The database merging track definition requires participants to report the size of data structures built from training data and the amount of data from the component databases that is used at run time to decide how many documents to retrieve from each database. We assume there is no interaction with the databases at run time to decide how many documents to retrieve, so the latter amount is zero for both the QC and MRDD merging strategies.

The QC merging technique must store the cluster centroids and the weight assigned to each cluster for each database. The cluster weights are completely dominated by the size of the centroid vectors. In our experimental environment, we do not store the centroid vectors themselves, but instead store the query vectors and recompute the centroids each time. The SMART vectors for queries 1-200 are approximately 800,000 bytes per database.

The MRDD merging technique has greater space requirements. The MRDD method must store an inverted file and dictionary for the collection consisting of the queries plus a list of the ranks of the relevant retrieved documents for each training query in each database. Our experimental setup (accidentally) uses a much larger than necessary dictionary and inverted file for the query collection. However, the inverted file for the queries contains 8,612 entries, so, assuming a 16 byte entry size, the inverted file would require at least 137,792 bytes. The dictionary contains 5828 terms, so its size would need to be at least $(5828 * 8) +$ *sum of the lengths of the character strings* bytes. The size of the data structure containing the ranks of the relevant retrieved documents obviously depends on the number of queries for which there is training data and the number of relevant documents per training query. The size of the relevance data for AP88 — a database that has relevance assessments for all 200 training queries and a larger than average number of relevant documents — is approximately 115,000 bytes.

The MRDD method requires more processing time than the QC method in addition to having greater space requirements. MRDD must solve an optimization problem each time it executes a query, while the QC method only needs to do a simple best match search in each collection to find the appropriate centroid and then a few arithmetic operations on the returned weights to compute the final number of documents to retrieve. However, since neither method has to communicate with the component databases to decide how many to retrieve, and since the computation of how many documents to retrieve will likely eliminate most databases from consideration, both methods are likely to be sufficiently quick in practice.

5 Conclusion

TREC-4 provided an opportunity to test our two database merging strategies on a new set of queries and, for the first time, in an environment where there were different amounts of training data for different databases. As in previous experiments, the effectiveness of the merged results was within 15% of the effectiveness of a single collection run when evaluated at moderate numbers of retrieved documents. The lack of relevance assessments for some queries in some databases had no obvious effect on the performance of the merged runs, although such an effect might be difficult to discern.

The merging strategies we use are isolated merging strategies in that they require no data from the component databases at runtime to decide how many documents to retrieve from each database. This makes the strategies efficient and suitable for use in environments where there is no central authority. A 15% degradation only amounts to approximately one fewer relevant document retrieved per query, and is thus quite reasonable when other circumstances prevent a single collection search.

These experiments raise issues that the results do not address and that therefore need further investigation. A major open issue for the isolated merging techniques is how the available training data affects the merging behavior. In settings other than TREC, one would expect many more training queries, but each query would have relevance data for only a few collections. The MRDD strategy may be more practical in such an environment since it is more dependent on quality training data. A possible alternative to the current strategy of using all available training data would be to select (by hand) a smaller number of exemplar queries. This would increase the efficiency of both the QC and MRDD methods, although its effect on the quality of the searches is unclear. Finally, Topics 202–250 are quite short as compared to previous TREC topics, and the MRDD method appears to have some difficulty with them. Could MRDD cope if all the training queries were as short?

A second issue involves the kinds of distinctions among databases a practical isolated merging strategy can be expected to learn. In TREC-4, the set of documents was divided into databases such that several databases were from the same source (e.g., WSJ90, WSJ91, and WSJ92). That is, the criteria that were used to classify documents into databases included considerations other than subject matter, which is likely to occur in other environments as well. While this does not appear to have been much of an impediment to the merging strategies in TREC-4, there may be an effect that is masked by the AP bias.

References

[1] Chris Buckley. Implementation of the SMART information retrieval system. Technical Report 85-686, Computer Science Department, Cornell University, Ithaca, New York, May 1985.

[2] Chris Buckley, Gerard Salton, James Allan, and Amit Singhal. Automatic query expansion using smart: Trec 3. In Donna K. Harman, editor, *Overview of the Third Text REtrieval Conference (TREC-3) [Proceedings of TREC-3.]*, pages 69–80, April 1995. NIST Special Publication 500-225.

[3] James P. Callan, Zhihong Lu, and W. Bruce Croft. Searching distributed collections with inference networks. In Edward A. Fox, Peter Ingwersen, and Raya Fidel, editors, *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 21–28, July 1995.

[4] Geoffrey Towell, Ellen M. Voorhees, Narendra K. Gupta, and Ben Johnson-Laird. Learning collection fusion strategies for information retrieval. In *Proceedings of the 12th Annual Machine Learning Conference*, July 1995.

[5] Ellen M. Voorhees, Narendra K. Gupta, and Ben Johnson-Laird. The collection fusion problem. In Donna K. Harman, editor, *Overview of the Third Text REtrieval Conference (TREC-3) [Proceedings of TREC-3.]*, pages 95–104, April 1995. NIST Special Publication 500-225.

[6] Ellen M. Voorhees, Narendra K. Gupta, and Ben Johnson-Laird. Learning collection fusion strategies. In Edward A. Fox, Peter Ingwersen, and Raya Fidel, editors, *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 172–179, July 1995.

Proximity Operators - So Near And Yet So Far

David Hawking and Paul Thistlewaite
Co-operative Research Centre For Advanced Computational Systems
Department Of Computer Science
Australian National University
{dave,pbt}@cs.anu.edu.au

December 6, 1995

Abstract

Testing of the hypothesis that good precision-recall performance can be based entirely on proximity relationships is a focus of current TREC work at ANU. PADRE's "Z-mode" method (based on proximity spans) of scoring relevance has been shown to produce reasonable results for hand-crafted queries in the Adhoc section. It is capable of producing equally good results in database merging and routing contexts due to its independence from collection statistics. Further work is needed to determine whether Z-mode is capable of achieving top-flight results. A new approach to automatic query generation designed to work with the shorter TREC-4 queries produced reasonable results relative to other groups but fell short of expectations based on training performance. Investigation of causes is still under way.

1 Introduction

The Parallel Document Retrieval Engine (PADRE) has previously demonstrated that full text scanning methods supported by parallel hardware permit powerful query constructors and rapid response to changing document collections [5, 6]. Since then, the addition of parallel-disk-resident inverted file indexes and dictionaries has potentially extended data handling capacity to the terabyte level, with an expectation of reasonable response times but some loss of flexibility [7]. Recent work [8] has added an `http` interface and enabled multiple simultaneous user sessions.

PADRE derives from an earlier ANU parallel system called PADDY [3, 12] which included a partial emulation of the functionality of the PAT text search program developed at the University of Waterloo [1, 2].

From an Information Retrieval perspective, PADRE now allows greater choice of relevance estimation functions (including one based on proximity spans [1]), is capable of simulating the operation of boolean style queries and provides a means of restricting the effect of precision-enhancing operations to only those documents which pass a test of basic relevance.

TREC runs based on the new improved PADRE were submitted in the Automatic Adhoc, Manual Adhoc and Database Merging categories. As was the case last year, official runs did not make use of inverted files but instead used full-text scanning methods over memory-resident data. A 512-node Fujitsu AP1000 (8 gigabytes of RAM) in Kawasaki was used for the Manual Adhoc submission. ANU's 128-node AP1000 (2 gigabytes of RAM) was used for most training

[1] Very similar to the one described in the University of Waterloo paper in these proceedings. Despite the fuzzy ancestral relationship described above, the two relevance models were developed completely independently. In fact the version of PAT with which we are familiar includes no concept of relevance at all.

runs and for the Automatic Adhoc and Database Merging runs. Previous training runs having suggested that merging results from CD2 with separately obtained results from CD3 did not introduce significant distortion, this approach was used for training and in the Automatic Adhoc submission.

The development at ANU of an experimental parallel file system [11], capable of loading a gigabyte of data in 20 seconds, has dramatically increased the number and scope of experiments which can be conducted in a given time.

The present paper describes the new PADRE features which underpin our submissions, documents the query generation processes and reports and analyses the results obtained. In addition, some further related experiments are reported.

2 Recent Extensions To PADRE Query Language

Since TREC3, a number of extensions and refinements to the query engine have been implemented.

2.1 Choice Of Term-Weighting Schemes

The query writer now has considerable freedom to select variants of the well-known *tf.idf* weighting scheme.

A document's estimated relevance R_d is still computed according to the following formula:

$$R_d = \sum_{t=1}^{k} (I_t \times r_{(t,d)}) \tag{1}$$

where:

k is the number of terms,

I_t is the manually assigned importance of term t, and

$r_{(t,d)}$ is the relevance of a document d due to term t.

but there are now a number of choices for the the calculation of $r_{(t,d)}$, such as:

$$r_{(t,d)} = \frac{f_{(t,d)}}{\sqrt{F_t \times l_d}} \tag{2}$$

$$r_{(t,d)} = \frac{f_{(t,d)}}{\log (F_t \times l_d)} \tag{3}$$

$$r_{(t,d)} = \frac{f_{(t,d)}}{\log F_t \times \log l_d} \tag{4}$$

$$r_{(t,d)} = \frac{f_{(t,d)}}{\log l_d} \tag{5}$$

$$r_{(t,d)} = 1.0 \; if \; t \; is \; present \; in \; d, \; otherwise \; 0.0 \tag{6}$$

where:

$f_{(t,d)}$ is the frequency of term t in document d,

F_t is the frequency of term t in the entire collection, and

l_d is the length of document d

F_t may optionally be defined as the number of documents containing term t rather than the total number of term occurrences.

2.2 Mandatory Inclusion Or Exclusion of Documents

PADRE now records *include* and *exclude* flag bits for each document in addition to accumulated positive relevance, accumulated negative relevance and maximum individual contribution to positive relevance. Accumulated document scores may be reset at any time without affecting the flag bit settings.

Commands are provided to set the flags for each element of the document set (or its complement) containing one or more members or the current match set. Documents may also be excluded from the relevant set on the basis that positive evidence is too weak, negative evidence is too strong or that positive evidence comes too heavily from a single source. Similarly, documents may be mandatorily included on the grounds that negative evidence is weak or that positive evidence is strong.

The exclude bit can be used to implement a type of query in which recall-oriented terms are used to define a *universe of discourse* set of documents prior to searches for precision-enhancing terms. Documents outside the universe of discourse remain excluded no matter how many occurrences of the latter terms they may contain. As an example, the universe of discourse may be defined as those documents containing at least m from a basic set of n key terms (or all of them). The precision phase searches for terms which do not by themselves imply relevance but whose presence in a document passing a general test of relevance increases the probability that it is actually relevant. For example in topic 204, the recall phase could identify documents dealing with *nuclear power plants* and the precision phase could boost the scores of documents in this universe of discourse which contain references to US-related terms and terms such as *gigaWatt.* Occurrence of such terms outside the universe of discourse is unlikely to be significant.

2.3 PRELATE - A GUI For Query Generation

Last year's Manual Adhoc submission was plagued by structural and other errors in the queries. This year, a prototype graphical user interface was built using TCL/Tk in an attempt to avoid this problem by making very evident the structure of complex queries and by rendering certain types of error impossible and detecting and warning of others. This interface is known as PRELATE (the Padre REtrieval LAnguage Topic Editor).

In general, PADRE queries are hierarchical, as may be seen in the sample PRELATE screen dumps below.

PRELATE proved to be quite effective in composing complex manual queries. It was useful but not infallible in avoiding errors. It is however, not presently designed to support interactive queries and would need significant modification for effective use in this role. A future line of development may convert the tool into an *applet* for use with a TCL/Tk web browser.

3 Manual Query Generation

3.1 Philosophy

Last year's manual PADRE queries were constructed with heavy reliance on PADRE proximity relationships. However, key individual terms with low manually assigned weights were sometimes used to improve recall, out of fear that the proximity relationships might be too seldom satisfied. Document relevance was calculated by summing *tf.idf* style weights derived both from singleton terms and from combinations of terms in proximity. This year, it was decided to pursue the proximity theme to its logical conclusion by basing relevance scores entirely on proximities.

Proximity relationships between terms intuitively seem to offer the potential to reduce spurious hits. To illustrate the point, an article was posted to the network news two years ago (and retrieved by an essentially boolean query) which contained widely separated occurences of the terms `Hawking`, `text` and `retrieval`. Its content bore no connection with one of the present authors and none with the field of IR. Had a proximity requirement been enforced, the document would not have been retrieved.

We adopted the following working hypothesis:

> *The closer together a set of intersecting terms, the more likely they are to indicate relevance.*

Proximity relationships can help to disambiguate different senses of the same word without the expense of semantic analysis. The word *bank* occurring in close proximity to *river, water, bridge* etc. is considered less likely to mean a financial institution than it would in other contexts.

Reliance on proximity has some of the flavour of passage-level approaches, but is cheaper to implement as it involves no semantic analysis of the text.

This year's Manual Adhoc submission was constructed entirely on the basis of *concept*s in proximity. No singleton term counted anything directly toward the relevance of documents. The presence of a concept in a document was signalled by a match against one of a [frequently large] set of alternative terms used to define it. These term sets were expanded considerably in order to improve recall. A proximity relationship between concepts will be referred to as a *concept intersection.*

3.1.1 Illustration of *Concept Intersections*

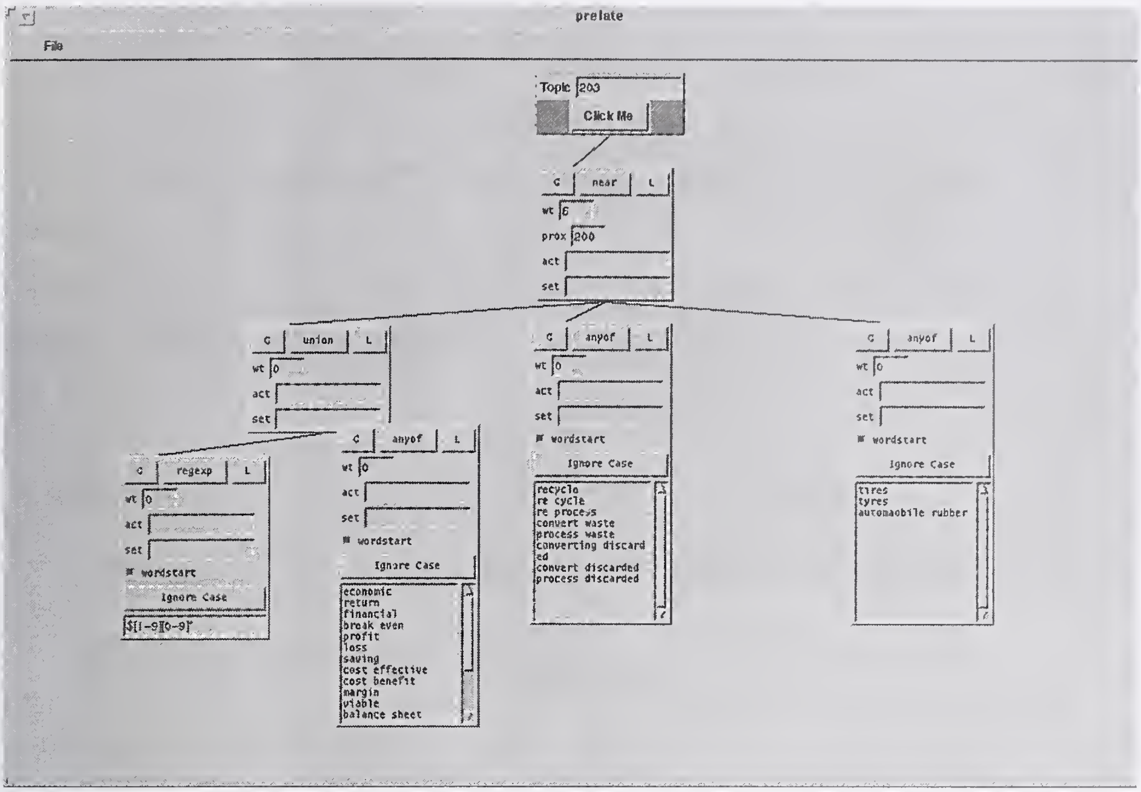
Let us use topic 203 ("What is the economic impact of recycling tires?"), to explain by example the process of generating queries based on concept intersections.

First, the topic was examined and three concepts were identified:

1. economic impact
2. recycling
3. tires

Words, phrases and regular expressions connoting each of the concepts were then generated from the query-writer's head, trying to take into account all the linguistic forms which might have been used to express the concept, including alternative spellings, plurals, different parts of speech and even mis-spellings.

As figure 1 shows, each set of terms connoting a concept were combined in an `anyof` operation. In the case of the *economic impact* concept, the match set arising from the regular



Topic 203

near: wt 5, prox 200

union: wt 0

regexp: wt 0

```
$[1-9][0-9]*
```

anyof: wt 0

```
economic
return
financial
break even
profit
loss
saving
cost effective
cost benefit
margin
viable
balance sheet
```

anyof: wt 0

```
recycle
re cycle
re process
convert waste
process waste
converting discard
ed
convert discarded
process discarded
```

anyof: wt 0

```
tires
tyres
automaobile rubber
```

Figure 1: Screen dump of PRELATE representation of PADRE query for topic 203 relating to the economic impact of recycling tires.

expression searching for dollar amounts is combined with the results of the anyof applied to the other simple terms using a set union operation.

The presence of one of the concepts by itself is not strongly indicative of relevance. Even a document containing many occurrences of many terms from only one of the concepts is not much more likely to be relevant. For example, a document containing many references to balance sheets, dollar amounts, profits etc. but nothing at all relating to tires or recycling is unlikely to be relevant but would score highly in a uniform simple-term weighting scheme.

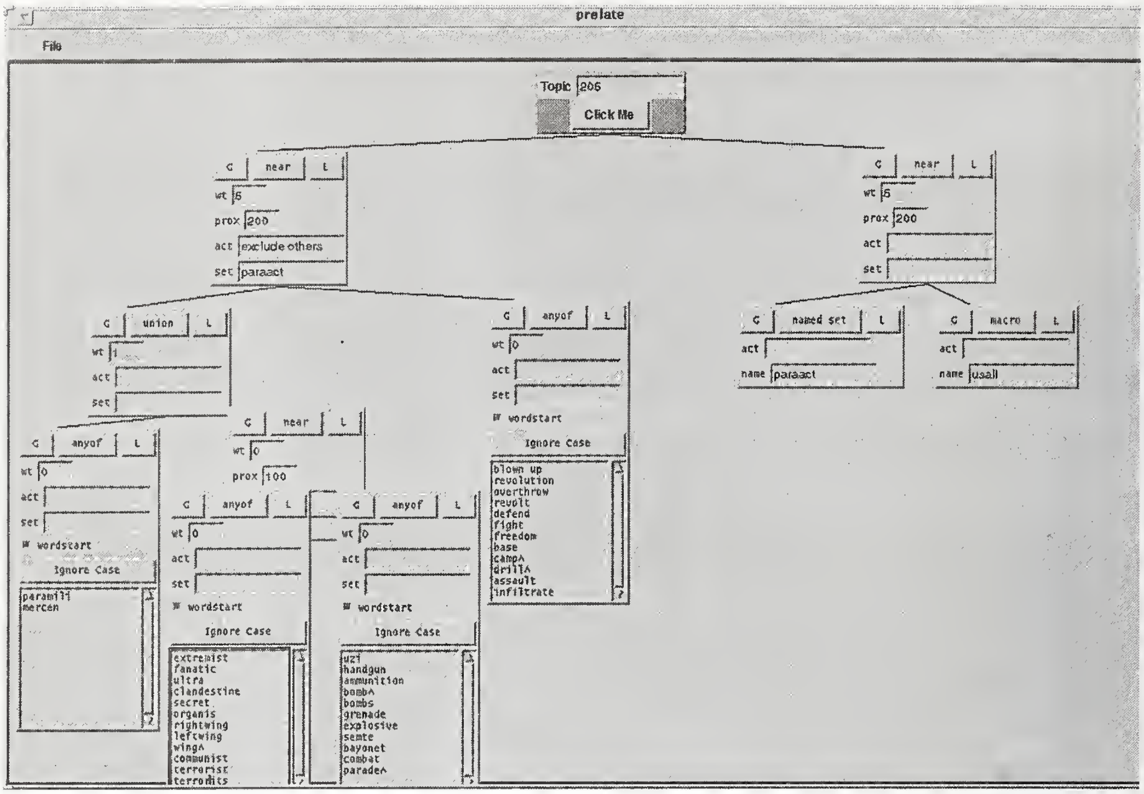
We postulated that the co-occurrence of all of the concepts within close proximity (a concept intersection) would be highly suggestive of relevance. Figure 1 shows that the concept intersection for query 203 is implemented using a near operator with a proximity of 200 characters and a weight of 5. All other operations have a weight of zero.

3.1.2 More Complex Proximity Relationships

Some topics gave rise to query structures involving multiple proximity relationships. This occurred in the following circumstances:

1. Cases where proximity relationships with small range were used to find "loose phrases" within a concept.

2. Cases in which a number of distinct proximity relationships independently imply relevance. That is, the presence of any one of the relationships $near(C_1, ..C_k)$, $near(C_{k+1}, ..C_l)$, ... suggests relevance.



```
Topic 206

near
wt 5
prox 200
act exclude others
set paraact

union
wt 1
act
set

anyof
wt 0
act
set
paramili
mercen

near
wt 0
prox 100

anyof
wt 0
act
set
extremist
fanatic
ultra
clandestine
secret
organis
rightwing
leftwing
wing^
communist
terrorist
terrorits

anyof
wt 0
act
set
uzi
handgun
ammunition
bomb^
bombs
grenade
explosive
semte
bayonet
combat
parade^

anyof
wt 0
act
set
blown up
revolution
overthrow
revolt
defend
fight
freedom
base
camp^
drill^
assault
infiltrate

near
wt 5
prox 200
act
set

named set
act
name paraact

macro
act
name usall
```

Figure 2: Screen dump of PRELATE representation of PADRE query for topic 205 relating to paramilitary activities in the U.S.A. The `usall` macro loads a pre-existing file of PADRE commands which define the U.S.A. concept in terms of alternative forms of the country name, state names, state capital names, names of geographical features, postal abbreviations, official state abbreviations and names of other prominent cities.

3. As for case 2, but where the same concept (tagged as a PADRE named set) may appear in more than one proximity relationship.
4. Cases in which an initial concept intersection, used to define a set of potentially relevant documents, itself becomes one component of a more restrictive concept intersection.

Figure 2 shows an example of multiple proximity relationships and the use of named sets.

3.2 Term Expansion Techniques

The expansion of terms to define a concept was done essentially by human free association supplemented by selective reference to the WordNet thesaurus, discussion with subject experts on some topics and by occasional references to a full lexicon of CD2/3. The latter was used to check whether a term occurred at all in the database, how it was spelled (eg. in Australian/British fashion as well as U.S.), and whether it was frequently mis-spelld in wayz which could be profitably included as search terms.

PADRE does not perform stemming. However, painful enumeration of all the different forms of a word was avoided through use of PADRE's ability to match arbitrary strings (optionally anchored at word starts). Regular expressions were used to locate dollar amounts in several queries.

3.3 Z-mode - Relevance Calculation Based On Proximity Spans

A relevance calculation method called, for want of better inspiration at 3am one morning just prior to the submission deadline, *Z-mode* was developed to more accurately reflect our working hypothesis (above). For each proximity relation instance discovered, the span of the relationship was calculated and the reciprocal of the span was added to the accumulated relevance score. The span is the length in words of the minimum substring of the text containing the instance of the proximity relationship.

Formula 1 above must be modified slightly to accommodate Z-mode. A document's estimated relevance R_d is now computed according to the following formula:

$$R_d = \sum_{t=1}^{k}(I_p \times r_{(p,d)}) \tag{7}$$

where:

k is the number of different proximity relationships in the query,

I_p is the manually assigned importance of proximity relationship p, and

$r_{(p,d)}$ is the relevance of a document d due to the specific proximity relationship p.

and:

$$r_{(p,d)} = \sum_{i=1}^{j} \frac{1}{S_i - 1} \tag{8}$$

where:

j is the number of instances of the proximity relationship found in document d, with unique starting point, and

S_i is the span of the ith instance.

For example, when searching the text below for a proximity relationship between *time, party,* and *people*, the italicised words constitute the first instance of such a relationship. The span of the relationship in this case is 15 and the contribution to the score of the document would be 1/14.

> The *time* has come for all good *people* to come to the aid of the *party.* We look forward to a time in which the people may party ...

The alert reader will notice that, assuming a typical proximity limit of 200 characters or so, the above example contains more than one instance of the sought-after relationship, including several with the same starting point. Overlapping instances are considered distinct provided they have unique starting points. Usually only the shortest of several spans sharing a common starting point is counted as an instance but, in some experiments, the longer ones were allowed to influence the score.

Initial trials with Z-mode on the training topics showed encouraging results. Subsequently, a number of variations on formula 8 were tried. Alternative numerators such as the total number of distinct proximity instances sharing the same start point or, alternatively, the same thing divided by the number of terms in the relationship were tried. Alternative denominators which reduced the rapid cutoff of the reciprocal were also investigated.

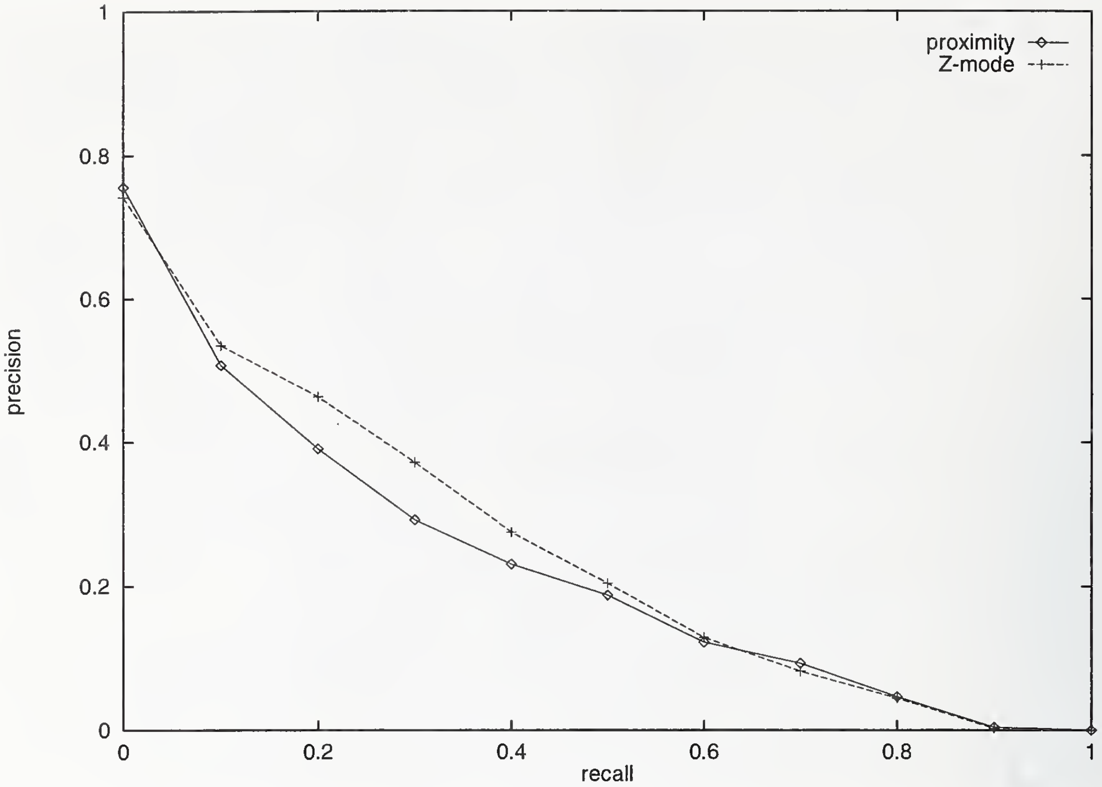


Figure 3: Precision-recall curves for PadreM1 and PadreZ.

On a dozen training topics, all worked better than the old non-Z-mode method but best results were achieved using the following formula:

$$r_{(p,d)} = \sum_{i=1}^{j} \frac{1}{\sqrt{S_i - 1}} \tag{9}$$

Note that Z-mode calculations of relevance do not involve collection frequency.

3.4 Experiment 1: PadreM1 (Initial Proximity-Only Run - Not Submitted)

A set of queries was written using the techniques described above and run using the weighting formula 3. Average precision over topics 202 - 250 was **0.2158**. The precision recall-curve is shown in figure 3.

3.5 Experiment 2: PadreZ (Official Manual Adhoc Run)

The padreM1 queries were manually modified in a systematic way. The Z-mode relevance function (formula 9) was selected. *Near* relationships searching for weightless loose phrases were left untouched but those which potentially contributed to relevance scores were changed to *znear* and the proximity limit was generally increased from 200 to 1,000 characters. This new set of queries, very strongly related to padreM1, was called padreZ.

Note that the padreZ results are significantly better than those for padreM1. Average precision has increased by 10% to **0.2383** and the total number of relevant documents retrieved has risen by 8%, from 3,326 to 3,602.

Compared with results obtained by other adhoc participants, padreZ:

- performed better than the worst for all measures on all topics.
- performed better than or equal to the median for:
    1. Rel. Retr. @ 100 on 32/49 topics
    2. Rel. Retr. @ 1000 on 23/49 topics
    3. Average precision on 26/49 topics
- achieved best performance for:
    1. Rel. Retr. @ 100 on 4/49 topics
    2. Rel. Retr. @ 1000 on 1/49 topics
    3. Average precision on 3/49 topics

3.6 Experiment 3: PadreW (Official Database Merging Run)

The padreZ queries were used unchanged in the database merging task. The ten subcollections were processed separately and the results combined by merging, sorting on relevance score, reranking and applying the 1,000 document cutoff.

As expected, the precision-recall results obtained are identical to those for padreZ. The only differences in the lists of documents retrieved were due to different orderings of equally ranked documents. Between the two lists of documents retrieved for all topics (each totalling 31,369 documents) only 3 differences were found. These all corresponded to a single query which produced a large group of equal scores around the 1,000 document mark.

3.7 Experiment 4: Enforcing a U.S.A. Context

At least five topics required rejection of otherwise relevant documents which did not relate to the U.S.A. In padreM1 and padreZ runs, this context was enforced by requiring the satisfying of (or increasing scores of documents which satisfied) an additional proximity test involving a very elaborate *usall* concept as described in figure 2.

In order to ascertain whether the considerable extra computational effort was justified by improved precision, the padreZ queries for the five above-mentioned topics were modified to remove the U.S.A. restriction and the results for the five new queries were compared with those for the originals.

When U.S.A. context was not enforced, average precision for the five queries fell from **0.3117** to **0.2840** and a significant difference in the precision-recall graphs may be seen in figure 4.

4 Automatic Query Generation

4.1 Experiment 5: PadreA (Official Automatic Adhoc Run)

The results for the automatic query generation run (padreA) were significantly inferior to our expectations, given the performance achieved during development against the TREC-3 topics and assessments.[2] Initial indications are that this poor performance was due, at least in part,

[2] Note that the TREC-3 topics were first translated to the terse TREC-4 style.

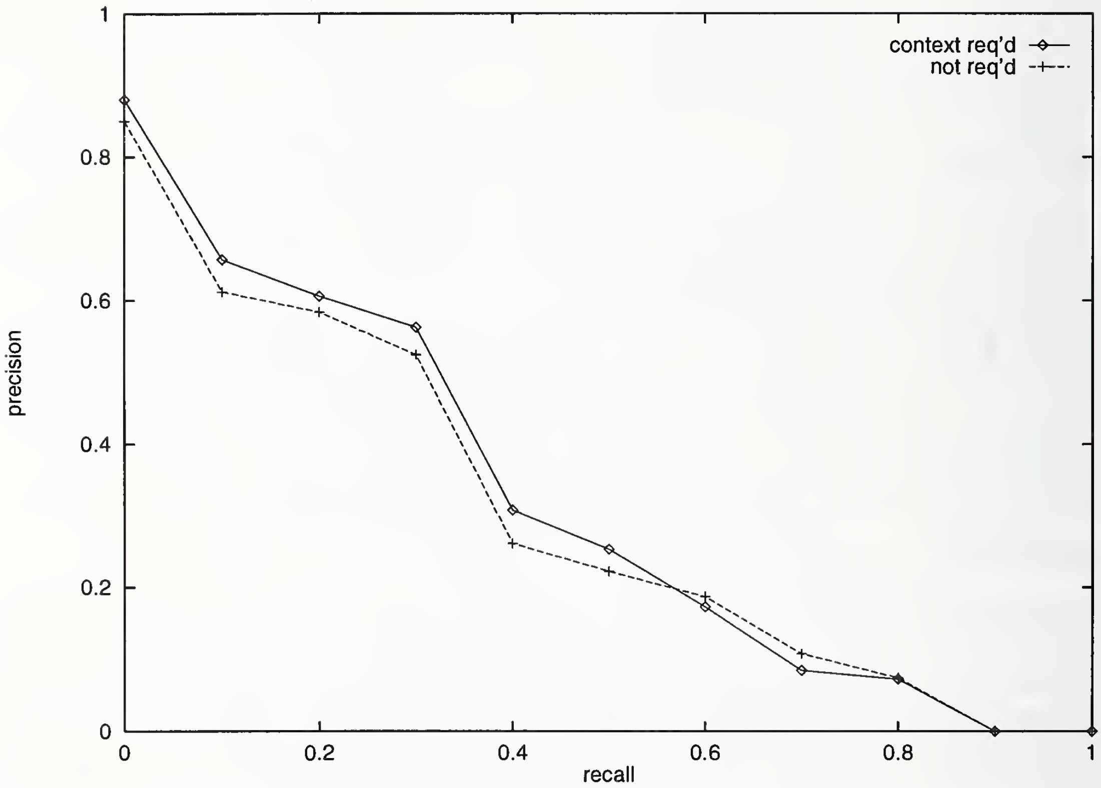


Figure 4: Precision-recall curves for queries 204, 205, 210, 227 and 235, showing the effectiveness of the method used to require a U.S.A. context.

to bugs in the mapping from the internal representation of the topic-query structure to the final PADRE query. The discussion of our automatic query generation approach will perforce focus on what was attempted rather than on what was achieved.

As with the padreZ run, our focus was on seeing to what extent proximity could be used as the primary ranking feature, rather than frequency.

The structure of automatically generated queries was two-fold: firstly, a set of anchoring terms was identified which was intended to define a subdomain on the document base with sufficiently high recall; then, subqueries derived from other parts of the topic were addressed to this subdomain of the document base, the purpose of which was to achieve better precision. Note however that the padreA queries used PADRE's standard `near` operator, rather than the `znear` used by padreZ, in order to bind these subqueries to the initial anchoring terms.

The terms in the padreA queries were restricted to being words or appropriate stems of words occurring in the topic. Additional capabilities permitting query expansion, using terms derived from relevance feedback and Wordnet thesaurus lookup, were not completed in time for inclusion. This was not considered to be a serious limitation to the experiment, as the primary motivation for query expansion would (we now believe incorrectly) be assumed to be to improve recall, and analysis of the TREC-3 topics and relevance judgements had shown that 99.4% of relevant documents contained at least one of the significant terms from the original topic. (A similar analysis of the TREC-4 data and gives a figure of 97.5%. A full discussion of these results will be the subject of another paper.) In retrospect, however, this restriction appears to be a significant limitation.

The poor performance of padreA manifested itself in two ways. In some cases, the software selected a set of anchoring terms that was too strict, and recall was low. In others, while initial recall was good, the subsequent subqueries did not make a sufficient improvement in precision.

Low initial recall was due to the software selecting the wrong terms from the topic to use as the anchor. This was in part due to the unwarranted assumption that the anchoring expression should be formed from a proper subset of the significant terms in the original topic. (The assumption is unwarranted because although a relevant document nearly invariably contains at least one word from the original topic, it need not of course be the same word for all documents relevant to the topic, and indeed in some cases best recall can not be achieved unless all significant words from the topic are incorporated into the anchoring expression.)

The second problem - poor precision - was probably due mostly to the restriction to using only terms contained in the original topic. Once our query expansion modules are complete, we should be in a better position to confirm this.

5 Discussion And Conclusions

Z-mode's total independence of collection statistics make it a very useful technique for database merging and routing, *provided* that:

- Z-mode queries are capable of achieving good precision-recall results, and
- Practical means for generating good Z-mode queries can be found.

On the first point, this year's Manual Adhoc results suggest that Z-mode queries are at least capable of achieving results which are better than average. Whether the reason they fall short of the very best relates to a fundamental limitation of Z-mode or whether it merely shows a lack of skill in query generation cannot be answered with present data. With the co-operation of the University of Waterloo we hope to investigate this question by converting their queries appropriately and processing with PADRE.

On the second point, manual generation of the queries was a time-consuming task, even with the help of PRELATE. We are optimistic that future ANU manual runs will involve only very small-scale manual intervention in an otherwise automatic process.

Acknowledgements

Fujitsu Laboratories provided access to AP1000 machines in the Fujitsu Parallel Computing Research Facility and assisted in various ways. Robin Stanton gave support and advice. Peter Bailey, Andrew Tridgell and David Campbell have worked on elements of the PADRE system. GNU regular expression code from the Free Software Foundation is incorporated in PADRE. We extend our sincere thanks to all these people and organisations.

Special thanks are also due to our subject experts and query advisers: Peter Bailey, Rosalie Balkin, Alan Connolly, Jo Evans, Michael Green, Sandy Gordon, Harold Griffiths, Kathy Griffiths, Ken Pogson, Maria Poulis, Tom Sutton, Andrew Tridgell, and David Westcombe.

The work reported has been supported by the Co-operative Research Centre for Advanced Computational Systems (ACSys).

References

[1] Fawcett, H. *PAT 3.3 User's Guide.* University of Waterloo Centre for the New Oxford Dictionary, Waterloo, Ontario (Feb 1991).

[2] Gonnet, G.H., Baeza-Yates, R.A. and Snider, T. *Lexicographic indices for text: Inverted files vs. PAT trees.* Report OED-91–01, University of Waterloo Centre for the New OED and Text Research, Waterloo, Ontario (Feb 1991).

[3] D.A. Hawking 'High Speed Search of Large Text Bases On the Fujitsu Cellular Array Processor,' *Proceedings of the Fourth Australian Supercomputing Conference*, pp. 83-90. Gold Coast, Australia (Dec 1991).

[4] D.A. Hawking and P.R. Bailey 'Towards a Practical Information Retrieval System For The Fujitsu AP1000,' in *Proceedings of the Second Fujitsu Parallel Computing Workshop*, paper P1-S. Kawasaki, Japan (Nov 1993).

[5] D.A. Hawking and P.B. Thistlewaite 'Searching For Meaning With The Help Of A PADRE', *Proceedings Of The Third Text REtrieval Conference (TREC-3)*, pp. 257-268. US Department of Commerce, NIST Special Publication 500-225 (Apr 1995).

[6] D.A. Hawking 'PADRE — A Parallel Document Retrieval Engine', in *Proceedings of the Third Fujitsu Parallel Computing Workshop*, paper P2-C. Kawasaki, Japan (Nov 1994).

[7] D.A. Hawking and P.R. Bailey 'A Document Retrieval Architecture Supporting Terabyte Collections', in preparation.

[8] D. Hawking, P. Bailey, D. Campbell, P. Thistlewaite and A. Tridgell 'A PADRE in MUFTI (A Multi User Free Text retrieval Intermediary)', in *Proceedings of the Fourth Parallel Computing Workshop* paper 26, Imperial College, London, (Sep 1995).

[9] Horie, T., Ishihata, H., Shimizu, T. and Ikesaka, M. 'AP1000 Architecture And Performance Of LU Decomposition,' in *Proceedings of the 1991 International Conference On Parallel Processing*, pp. 634-635. August 1991.

[10] Ishihata, H., Horie, T., Inano, S., Shimizu, T. and Kato, S. 'CAP-II Architecture,' in *Proceedings of the First Fujitsu-ANU CAP Workshop*, paper 1. Kawasaki, Japan (Nov 1990).

[11] A. Tridgell and D. Walsh 'The HiDIOS Filesystem' in *Proceedings of the Fourth Parallel Computing Workshop*, pp. 53-63. Imperial College, London (Sep 1995).

[12] *PADRE home page.* `http://cap.anu.edu.au/cap/projects/text_retrieval/`

TREC-4 Ad-Hoc, Routing Retrieval and Filtering Experiments using PIRCS

K.L. Kwok & L. Grunfeld

Computer Science Dept., Queens College, CUNY,
Flushing, NY 11367. email: kklqc@cunyvm.cuny.edu

ABSTRACT

Our ad-hoc submissions are pircs1 which is fully automatic, and pircs2 which involves manually weighting some terms and adding some new words to the original topic descriptions. The number of words added are minimal. Both methods involve training and query expansion using the best-ranked subdocuments from an initial retrieval as feedback. For our routing experiments we make use of massive query expansion of 350 terms in pircsL, with emphasis on expansion with low frequency terms. Training is done using short and top-ranked known relevant subdocuments. In pircsC, we define four different 'expert' queries (pircsL being one of them) for each topic by using different subsets of training document, and later combine their retrieval results into one. Filtering experiment is done with the retrieval lists of pircsL. For each query, we use the training collections to define retrieval status values (RSVs) where the utilities are maximum for the three precision types. These RSVs are then used as thresholds for the new collections. Evaluated results show that both ad-hoc and routing retrievals perform substantially better than median.

1. INTRODUCTION

The PIRCS retrieval system has been described in our previous TREC papers [KwPK93,KwGr94b,KwGL95] as well as in [Kwok95,90]. Basically, given a query q_a and documents d_i, we use two retrieval algorithms that give the following document-focused and query-focused ranking RSVs respectively:

$$RSV_{i/d} = \sum_k w_{ik} * w_{ka}$$
$$RSV_{i/q} = \sum_k w_{ak} * w_{ki}.$$

w_{ka} is the weight of term t_k used in q_a and is content oriented, and w_{ik} is the weight assigned to t_k based on its collection properties and is more discrimination oriented. (Similar for w_{ki} and w_{ak}). They are implemented as weighted edges connecting documents and queries to terms in a bi-directional network. The sum is over all terms common to d_i and q_a. Results of these two algorithms are then combined to return a final ranking RSV for document d_i as follows:

$$RSV_i \quad = a*RSV_{i/d} + (1-a)*RSV_{i/q}$$

where a<1 is an adjustable parameter usually set to between 0.7 and 0.9. By viewing a document as not monolithic but composed of conceptual components each representable as a term and hence removing the 'binary' assumption of the normal probabilistic model, w_{ka} is given by $S(TermFreq_{ak}/Length_a)$, and correspondingly for w_{ki}. S(.) is a signal function to suppress outlying values and $Length_a$ is the length of q_a including repeating terms [Kwok90,95]. w_{ik} (and correspondingly for w_{ak}) consists of a trainable factor and an Inverse Collection Term Frequency factor ICTF = log [(TotalTerms-ColFreq$_k$)/ColFreq$_k$]. ICTF is like the Inverse Document Frequency except that it also accounts for term frequencies, not just 'binary'. The trainable factor at the initial stage when queries and documents perform hard self-learning would be given by log [TermFreq$_{ik}$/(Length$_i$-TermFreq$_{ik}$)]. In our system we perform soft self-learning and w_{ik} has value lying between ICTF and ICTF + the hard self-learning value. When more documents known relevant (or assumed relevant) to q_a are available, this trainable factor can attain more accurate value and provide better retrieval results. Moreover, based on the training documents, edges connecting q_a to new terms may be added resulting in an expanded query. This enables retrieval of documents that do not share common terms with the original query. These processes of training, query expansion and retrieval are performed by sophisticated algorithms and are implemented as activation spreading in the network.

Sections 2 and 3 contain descriptions of our ad-hoc and routing experiments and in Section 4 the filtering experiment. Our conclusion is in Section 5.

2. AD-HOC RETRIEVAL EXPERIMENTS

2.1 Initial Queries

Ad-hoc experiments for TREC-4 differ from previous TRECs in that the new topic set Q5 (#202-#250) are deliberately short, consisting of one to three lines of text in the Description Section only. After automatic processing in our system with stopword removal and stemming, the number of unique terms left totaled 305, giving an average of only 305/49 = 6.22 unique stems per query. A distribution table of query sizes is shown in Fig.1 under pircs1. These small **initial** query sizes are not suitable for statistical systems like PIRCS which relies on accumulating evidence from many terms and phrases to define a topic of need. This has been shown to be true in [LuKe95]. Another related problem arising from these small query sizes is that the short descriptions generally do not offer opportunity for the query composers to repeat words, so that we also end up with queries having uniform, equally weighted terms with no differentiation of importance for representation. Since we will use query expansion for our **final** retrieval results (Section 3.2), the quality of the **initial** retrieval is important. In view of this, we decided to do three experiments: pircs1, 1a and 2, differing in the initial queries with which we start.

Pircs1 relies on initial queries obtained from the raw topics automatically via stemming and stopword removal. This gives the basis result from which we can try to improve. For pircs1a (unsubmitted), we manually weight the terms in the queries obtained in pircs1 by replicating some of the words. They are chosen based on our belief that these words represent concepts that are central to the topical needs. We are also careful to treat every replicated word separately so that we do not accidentally add 2-word adjacent phrases. In pircs2, we replicate words without the previous precaution, and also augment some queries with new words. They are manually added based on synonyms, acronyms, general-specific, or highly associated words that come to mind after reading the topics. No retrieval was involved. However, in keeping with the spirit of testing short queries, we only add minimally. After processing, a total of 52 terms are added to 30 queries, averaging 52/30 = 1.73 per (changed) query. 19 queries are left with the same words as in pircs1a. Generally we add to the shorter queries and at most three content words per query. These manual operations are different from those done in [BCCN95]: they expand the queries automatically first and then manually delete, re-group, re-weight some of the terms or add new terms from the narrative section; we re-weight and add terms prior to processing with no term deletion, and the query is later expanded automatically. The distribution of query sizes for pircs2 is also shown in Fig.1 with an average of 7.29 terms per query.

Number of Uniq.Terms	Number of Queries pircs1	pircs2
2	2	0
3	5	2
4	9	3
5	8	7
6	9	11
7	2	7
8	1	3
9	6	7
10	3	4
11	0	1
12	2	2
13	2	1
14	0	1
Average	6.22	7.29

Fig.1: Distribution of Query Sizes for Pircs1 and Pircs2

2.2 Disks 2 and 3 Textbase and Expanded Queries

Ad-hoc retrieval was done using the Tipster Disks 2&3 collections. They are separated into four subcollections with documents segmented into about 550-word chunks as in previous TRECs. They are served by a master lexicon of 643,755 unique terms that includes 55,599 entries of our semi-automatic 2-word phrases.

The queries discussed in Section 2.1 are used to do an **initial** retrieval on the textbase. The 40 best-ranked subdocuments of each query are then employed as 'relevant' documents to train and expand the initial queries automatically as in a routing situation. In

TREC-3, we used only 6 best-ranked items which was insufficient [RWJH95]. These 40 subdocuments of a query define a set of terms; the best 50 such terms are selected based on occurrence frequency (>=5) and the average probability of occurrence within the 40 documents. They are added to the initial query, resulting in an expanded query with average size of 52.4 terms. A second round of retrieval is done using the expanded queries, and the document ranking then constitutes our final results. This procedure is repeated separately starting with each of the three initial query types discussed in Section 2.1.

2.3 Results and Discussion

Results of our six experiments are summarized in Fig.2, where the * entries denote our official results. Percentage increases from the base values (%/%) are also shown horizontally (before and after query training and expansion) and vertically (between initial query types).

Initial Qry Type	Initial Retrieval	After Expansion Retrieval
pircs1 (auto)	3327/.2015 (%/%) (% / %)	*3896/.2599 (17.1 / 29.0) (% / %)
pircs1a (manual)	3642/.2259 (%/%) (9.5 / 12.1)	4258/.2828 (16.9 / 25.2) (9.3 / 8.8)
pircs2 (manual)	4044/.2619 (%/%) (21.6 / 30.0)	*4562/.3064 (12.8 / 17.0) (17.1 / 17.9)

Fig.2: Relev.Retr. @1000 / Non-Interpolated Precision Results Before and After Query Expansion for the 3 Initial Query Types Averaged over 49 queries.

Initial Retrieval Using Different Initial Query Types

It can be seen from the Initial Retrieval column of Fig.2 that by simply replicating some content words in the original topic statements improvements of 9.5% and 12.1% are obtained for relevants retrieved @1000 and the average non-interpolated precision (pircs1a 3642/.2259 vs pircs1 3327/.2015). Out of 49 queries, 38 led to equal or better results after weighting and 11 gave worse, a ratio of nearly 4:1. This operation is very simple and the choice of words are usually quite obvious with little intellectual effort because the topic descriptions are so short. With a good text editor this can be done in less than 1/2 minute per query. The recommendation is that users who submit short descriptions should be advised to replicate and append some content words, and this can buy a gain of roughly 10% from PIRCS.

When we also augment some of the topics with new words, larger gains of 21.6% and 30% in relevants retrieved @1000 and precision are observed (pircs2 4044/.2619 vs pircs1 3327/.2015). Out of 49 queries in pircs2, 41 lead to results equal or better than pircs1 and 8 worse, a ratio of 5:1. Thus, even though we add minimally, the effect on the initial retrieval is substantial. Finding words to augment the queries however takes more time than just selecting words to replicate, probably about 3 hours for 50 queries. Quite often, trying to think of new words to add and later abandoning the effort takes longer time than having obvious associated terms.

Retrieval Using Different Expanded Queries

From Fig.2 we observe that retrieval via our query expansion procedure using best-ranked subdocuments for training is quite successful, leading to improvements of 12.8% to 29.0% from the initial query results. The procedure for training and query expansion is fully automatic. Comparing horizontally, we see that when the basis is higher the improvement is less (e.g. pircs2 initial 4044/.2619 vs expansion 4562/.3064: 12.8%/17.0%; pircs1 initial 3327/.2015 vs expansion 3896/.2599: 17.1%/29.0%), which is not unexpected.

Comparing vertically, we see that if one starts with better initial queries that lead to better initial retrieval, the final query expansion retrieval is also better (pircs2 4562/.3064: 17.1%/17.9% better than pircs1 3896/.2599). Simply weighting the query content terms only produces about half the improvements of pircs2 from pircs1 (pircs1a 4258/.2828: 9.3%/8.8% better than pircs1 3896/.2599). Pircs2 has 32 queries better and 17 worse than pircs1, and for pircs1a 30 better and 19 worse in these expanded query retrieval results.

As noted above, the exercise of weighting and adding new words to queries does not always lead to better results. Examples of successes are: the added synonyms car, auto and automobile in queries 219, 230 and 237; rubber (associated with tire) in 203; coal (specific-general to fossil fuel) in 243. Examples of failures are: death sentence (synonym to capital punishment) in 222; militia group (associated with paramilitary) in 231; iron (associated with steel) in 218; newborn deaths (associated with infant mortality) in 215.

Using pircs2 results, we observe that it recalls (4562/6501) 70.2% of "all" relevants at 1000 documents retrieved. At 10 documents retrieved, one can expect more than 5 of them are relevant, and at 30 more than 13.

Comparison with Other TREC-4 Sites

Comparison with the MEDIAN values of TREC-4 submissions are summarized below in Fig.3. It can be seen that results of pircs1 and especially pircs2 substantially outperform the median. As before, we also calculate MAXI-retrieval as a hypothetical system that returns the best performance for each query among all sites. This assumes that we have an intelligent agent who is able to choose the best retrieval system among this set of participants for each query, and would reflect the best we can do using our collective wisdom at this time. This MAXI-system will return an average precision, precision at 100 docs and at 1000 docs of 0.4638, 0.4743 and 0.1115 respectively. Thus, pircs2 achieves 66.1%, 69.9% and 83.5% and pircs1 achieves 56.0%, 60.4% and 71.30% respectively of these best values. PIRCS appears to achieve good results at the high recall region.

	pircs1			**pircs2**		
	>	=	<	>	=	<
av. prec:	30(3)	3	16	40(6)	0	9
rel_ret @ 100:	32(3)	4	13	39(6)	1	9
rel_ret @ 1000:	39(8)	2	8	44(14)	2	3

(figure in parenthesis is number of queries equaling the best values)

Fig.3: Comparison of Ad-Hoc Results with the Median from All Sites

3. ROUTING RETRIEVAL EXPERIMENTS

3.1 Routing Queries

We submitted two routing retrieval results. The first one is PircsL, where L means large query. Following Buckley et.al. [BuAS94] we found that massive term expansion of queries is beneficial for our model too.

The second submitted retrieval, PircsC , involves an experiment with combination of retrievals. Combining queries has received a lot of attention lately. Lee tried combining different normalizations available in the SMART system [Lee95], Kantor combined natural language, hard boolean and inferential retrieval, (decision level fusion) using several different schemes [Kant95], Fox and Shaw combined different soft-boolean and vector retrievals [FoSh94], and we also combined soft-boolean with PIRCS retrieval in our previous TREC ad-hoc experiments. In TREC-4, our approach is different, in that we try to create queries by limiting the training data to certain collections. For example one set of queries was trained by using only relevant information from the Ziff and FR collections. The rationale for this is that certain collections may have better, more focused coverage of some topics. The other interesting idea is an attempt to predict the performance of each method for each query and combine them accordingly.

PircsL (Large Query)

For the TREC4 routing experiments our starting point was the methods developed for TREC3 [KwGL95]. The learning and retrieval algorithms remained the same. Documents are broken up into 550 word segments, and if more than one subdocument is retrieved, their retrieval status value is combined by the formula RSV=1.0 * first + .1 * second + .05 * third. Only relevant documents are used for training. Judged non relevants are ignored. Queries include terms from original query and terms expanded from relevants. Term weights are based on relevant documents.

Experiments after TREC3 revealed that our queries have very good recall, but the precision could be improved. The term selection formula was changed to favor low frequency terms. The term selection formula in TREC2 was sum (Wi / log [max (2000,DOCFREQi)])/N,

where Wi is the weight of term i in the relevant document defined as (count of term i)/(count all terms), N is the number of relevant documents, DOCFREQi is the document frequency of term i, the sum is over all relevant documents for that query. Note that this takes into account the within document frequency. The denominator favors higher frequency terms, which represent more general concepts. This was reasonable for the relatively small query expansion. For TREC3 we substituted 20 for 2000, ranking the low frequency terms much higher for selection. The submitted PircsL queries were expanded by 350 terms.

In the routing environment, where a large number of relevant documents of different sizes and quality are available, selecting a good subset for training queries is important, rather than using all. A number of strategies are described in [KwGr94]. Two methods were found to produce good results: (a) selecting short documents and (b) selecting the top ranked subdocument from each document.

(a) Selecting short documents has the advantage, that it does not require ranking, and the documents will not contain many other topics. Also documents that are not ranked high by our system will be included. The disadvantage is, that not all queries have enough short documents to produce a good query.
(b) Selecting top ranked subdocuments will give a more broad based sample to train on, but it is biased toward the systems own retrievals, and does not take full advantage of the information available from other relevants. In practice it is slightly better than selecting short documents.

For the pircsL query, we use a hybrid selection method. We selected all short documents (those with 550 words or less) and added to it the top 50 retrieved relevant subdocuments.

An important issue is the selection of the test collection. Creating the query first and testing them on the data on disks 1, 2 and 3 would not be appropriate, since that would be a retrospective retrieval, and good results there may not give good results on an unknown collection. In fact there is a danger of overfitting the test data. On the basis of the information available to us, we assumed that the routing retrieval will be on documents similar to the Ziff and Federal Register collections. Therefore we decided to evaluate the results, based on retrievals on the Ziff and FR collections on disk 2, and train the queries on the other collections. After the methods were selected, we added the Ziff and Fr relevants from disk 2 to the training documents, to create the official query.

PircsC (Combined Queries)

Combining retrievals are known to be beneficial, provided that the retrievals are independent and are about the same power. PIRCS has the ability to combine different methods in the network. The reasoning is, that relevant documents will occur more non-randomly, in the retrieval lists than irrelevants, therefore combining them could improve their ranking.

A person who reads only the Wall Street Journal will probably create a different query than a person who reads PC Magazine, given the needs and concepts contained in a topic. The hypothesis is, that the large number of training documents available from publications of differing perspectives will allow for the creation of independent query formulations by different 'experts'.

For pircsC we combined 4 different query formulations:

Expert #1: (pircsL) Long query, expansion 350. This is the same as pircsL.
Expert #2: (short) Short query, term expansion is only 80. The training method for this query favors larger frequency words. The training documents are the same as for #1. This query is very similar to our query at TREC-3.
Expert #3: (ZFD) Query was trained only on Ziff and Federal Register and DOE documents. Query expansion is 300.
Expert #4 (WAS) Query was trained only on WSJ AP and SJM documents, Query expansion 300.

There are a number of ways to add retrievals. A simplistic way is just to add the retrieval status value after some normalization, so that the rsv of different queries are compatible. In the routing environment we can get a reasonable estimate as to each method's performance for each query, and use this information to enhance the result.

Experimenting with combining two retrievals only, we found, that when they perform similarly, it is beneficial to add them, otherwise take the result of the better one. A function which has this desired behavior for large N is (Xi)**N, where Xi is the Av11 for the method i and N is an integer.

For the pircsC submitted query, we made 2 adjustments. Since for very low Av11 this expression goes very quickly to zero, we added a .1 to all Av11 values. Also, since estimates were made based on retrieval performance in Ziff and FR on disk2, the performance of the Expert #3 was probably underestimated, since it is missing 50 percent of it's training documents (since it was created using only disk1 at this point), therefore we added an extra 0.03 to all the av11 for this 'expert'. The value of N was 4.

3.2 Results

In all the results below we use the short query as the baseline, since it is closest to our TREC3 query, so we can assess any improvements on it.

	Rel Ret	**% chg**	**Avg prec**	**% chg**
short	5555	0.00%	0.3713	0.00%
pircsL	5645	1.62%	0.3901	5.06%
zfd	5501	-0.97%	0.3672	-1.10%
was	5406	-2.68%	0.3582	-3.53%

	Rel Ret	**% chg**	**Avg prec**	**% chg**
comb1111	5658	1.85%	0.3900	5.04%
comb8421	5626	1.28%	0.3924	5.68%
comb1	5658	1.85%	0.3926	5.74%
pircsC	5635	1.44%	0.3909	5.28%
comb8	5612	1.03%	0.3891	4.79%
comb200	5571	0.29%	0.3843	3.50%

Out of the 4 experts, the large query (pircsL) performed best, as was expected. It improved by 5% over the short query, which uses basically the same method that was submitted to TREC3. Although it is difficult to make comparisons accross different query collections, this seems to indicate, that the TREC4 queries are somewhat more difficult, since there the score was .388. The other 2 experts performance was worse. It is interesting to note that the ZFD expert, which was trained on a similar type of collection as the target did not do better. Perhaps the reason is that there were not enough training documents.

It is interesting to notice, that out of the 50 queries PircsL had 20 bests and the others about 10 each. This bears out the hypothesis, that while the individual experts may not be very good overall, they can become very good at individual queries.

A number of retrievals using different combinations were tried. All combination were combined using the equation RSVj= SUM (ai * RSVi j) , where ai is a constant for method i and RSV is the return status value for method i , for document j and the sum is over the methods i.

The different combinations vary the value of a. Comb1111 is what was called a simplistic addition, a=1. Comb8421 is defined as: a= 8 to the best expert a=4 to the second a=2 to the third and a=1 to the fourth. The idea is the we want the best retrieval to dominate and the other retrievals to provide corrections.

The other methods use the formula a = (Xi)**N , where Xi is the Av11 for the method i and N is an integer. Note that if N=1 then a=Av11 and if N=200 then the best retrieval has a=1 and the others 0, except in the case where the Av11 is very close to each other.

The other 4 combinations vary the N in the above formula. N=1,4,8 and 200 were tried.

The differences may not be significant to draw any conclusions. From the first 2 combinations it seems that its better to give more weight to the better expert. From the last 4 combinations we can conclude the opposite.

We still believe, that it is possible to create multiple experts from large number of relevant documents, perhaps by using more sophisticated methods, such as clustering. Also prediction of expert performance can be improved. Note that the combination with N=200, which takes the best performing expert for each query did worse the PircsL. Had we been able to predict the best expert for each query and use that query the result would be .4027, a 3% improvement.

Comparison with Other TREC-4 Sites

	pircsL			pircsC		
	>	=	<	>	=	<
av. prec:	41(3)	2	7	42(5)	0	8
rel_ret @ 100:	35(4)	8	7	44(6)	8	8
rel_ret @ 1000:	41(19)	7	2	41(19)	6	3

(figure in parenthesis is number of queries equaling the best values)

Fig.4: Comparison of Routing Results with the Median from All Sites

4. FILTERING EXPERIMENTS

We also participated in the filtering track. Our approach was to try to predict the rsv at which the filter evaluation function is the maximum. The training data used were the disk2 Ziff and FR document collections. We did another set using the entire disk2.

The retrieval was done using the pircsL query discussed above, therefore all training was done using retrospective retrieval. We think this has the effect of concentrating the relevants at the top of the retrieval, making the maximum filter evaluation point higher that what it should be, leading to a more conservative estimate, if there are a lot of relevent documents, the opposite may happen if there are only a few.

Training on the the entire disk 2, returned resulted at a much lower cutoff and more documents returned, but for the submission we decided to be conservative and submitted the result based on the Ziff and FR collections only.

At this point we do not have results on which method performed better.

5. CONCLUSION

The PIRCS retrieval system and its learning capabilities have consistently been demonstrated to give exemplary performance in ad-hoc and routing retrievals. In ad-hoc, in the absence of known relevants, we have shown that training and query expansion from a set of best ranked subdocuments from an initial retrieval is beneficial. The better the initial retrieval, the better the final results. Manually weighting terms in a given topic, and minimally adding a few new terms can lead to this better initial environment. In the future, we will try to automate these manual processes. For routing, we have shown that PIRCS can benefit from massive query expansion of over 300 terms. We also introduce a method of defining several 'expert' queries from a single topic based on subsets of the relevant documents. Combining the retrieval results from several 'expert' queries lead to slightly better results. More investigation needs to be done to find the best method for this operation.

ACKNOWLEDGMENTS

This work is partially supported by a PSC-CUNY grant.

REFERENCES

[BCCN95] Broglio, J, Callan, J.P, Croft, W.B & Nachbar, D.W. Document retrieval and routing using the INQUERY system. In: Overview of the Third Text REtrieval Conference (TREC-3). Harman, D.K. (Ed.). NIST Special Publication 500-225, 1995, pp. 29-38.

[BuAS94] Buckley, C., Allan, J. & Salton, G. Automatic Routing and Ad Hoc Retrieval using SMART: TREC2. In: The Second Text REtrieval Conference (TREC-2). Harman, D.K. (Ed.). NIST Special Publication 500-215, 1994, pp. 45-55.

[FoSh94] Fox, E. A and Shaw, J. A. Combination of multiple searches. In: The Second Text REtrieval Conference (TREC-2). Harman, D.K. (Ed.). NIST Special Publication 500-215, 1994, pp.243-252.

[Kant95] Kantor P.B Data Fusion in Information Retrieval. In: Overview of the Third Text REtrieval

Conference (TREC-3). Harman, D.K. (Ed.). NIST Special Publication 500-225, 1995, pp.319-332.

[KwGL95] Kwok, K.L, Grunfeld, L & Lewis, D.D. TREC-3 ad-hoc, routing retrieval and thresholding experiments using PIRCS. In: Overview of the Third Text REtrieval Conference (TREC-3). Harman, D.K. (Ed.). NIST Special Publication 500-225, 1995, pp.247-255.

[KwGr94a] Kwok, K.L. & Grunfeld, L. Learning from relevant documents in large scale routing retrieval. In: Proc. Human Language Technology Workshop, ARPA, Mar 8-11, 1994 Plainsboro, NJ. Morgan Kaufmann, San Francisco, 1994, pp. 358-363.

[KwGr94b] Kwok, K.L. & Grunfeld, L. TREC-2 Document retrieval experiments using PIRCS. In: The Second Text REtrieval Conference (TREC-2). Harman, D.K. (Ed.). NIST Special Publication 500-215, 1994, pp. 233-242.

[Kwok90] Kwok, K.L (1990). Experiments with a component theory of probabilistic information retrieval based on single terms as document components. ACM Transactions on Office Information Systems, 8:363-386.

[Kwok95] Kwok, K.L (1995). A network approach to probabilistic information retrieval. ACM Transactions on Office Information Systems, 13:325-353.

[KwPK93] Kwok, K.L., Papadopolous, L & Kwan, Y.Y. Retrieval experiments with a large collection using PIRCS. In: The First Text REtrieval Conference (TREC-1). Harman, D.K. (Ed.). NIST Special Publication 500-207, 1993, pp. 153-172.

[Lee95] Lee, J.H. Combining multiple evidence from different properties of weighting schemes. In: Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp.180-188, 1995.

[LuKe95] Lu, X.A & Keefer, R.B. Query expansion/reduction and its impact on retrieval effectiveness. In: Overview of the Third Text REtrieval Conference (TREC-3). Harman, D.K. (Ed.). NIST Special Publication 500-225, 1995, pp.231-239.

[RWJH95] Robertson, S.E, Walker, S, Jones, S, Hancock-Beaulier, M.M, Gatford, M. Okapi at TREC-3. In: Overview of the Third Text REtrieval Conference (TREC-3). Harman, D.K. (Ed.). NIST Special Publication 500-225, 1995, pp.109-126.

Research in Automatic Profile Generation and Passage-Level Routing with LMDS

Julian A. Yochum

Logicon, Inc.
2100 Washington Blvd.
Arlington, VA 22204-5703
jyochum@logicon.com

Abstract

This paper describes the development of a prototype system to generate routing profiles automatically from sets of relevant documents provided by a user, and to assign relevance scores to the documents selected by these profiles. The prototype was developed with the Logicon Message Dissemination System (LMDS) for participation in the Fourth Text REtrieval Conference (TREC-4).

Each generated profile contains two sets of terms: a very small set to select documents, and a much larger set to assign a relevance score to each document selected. The profile generator chooses each term and assigns a weight to it, based on its frequency of occurrence in the set of documents provided by the user, and on its frequency of occurrence in a large representative corpus of documents. The LMDS search engine uses the resulting profiles to select documents, and then passes the documents to the scoring prototype for ranking. The score assigned is a function of the weights of all profile terms found in the entire document, and of those found in fixed-length overlapping passages within the document.

Performance figures and TREC-4 results are included. An appendix describes a modification to the TREC-4 algorithm, made since the conference, which has produced significant improvements in both recall and precision.

1. Introduction

LMDS is a commercial off-the-shelf (COTS) product, designed specifically for high-speed document routing [1] on a wide range of hardware and operating system platforms. LMDS users create interest profiles to specify the types of documents that they wish to receive. Each profile contains a list of tokens (or "search terms") which may appear in such a document, together with a Boolean expression indicating the logical combination in which the tokens must occur for LMDS to select the document. In addition, the user may optionally attach a weight to each search term.

When LMDS selects a document, it passes the document to a scoring routine, together with pointers to the set of search terms discovered in the document, and to their associated weights. In addition, LMDS provides an API which allows any installation to customize the scoring routine for its own purposes. The resulting document scores are then used to order the documents for display.

A feature not yet available in LMDS is "query by example," the capability to generate profiles automatically, based on a sample of documents which the user deems relevant to a specific topic. Since the relevance judgments associated with the TREC-4 training documents provide 50 such sample sets, our objective for TREC-4 was to use these sample sets to refine and enhance the automatic profile generation and relevance ranking prototype system [2] that we developed for TREC-3.

LMDS is designed to run thousands of profiles against incoming documents in a minimum amount of time and with minimum hardware requirements. To qualify as workable enhancements to LMDS, therefore, any algorithms for automatic profile generation and relevance ranking must not only be effective, but must also be compact and fast. The TREC-4 prototype system was designed with these goals in mind, and a hardware configuration was

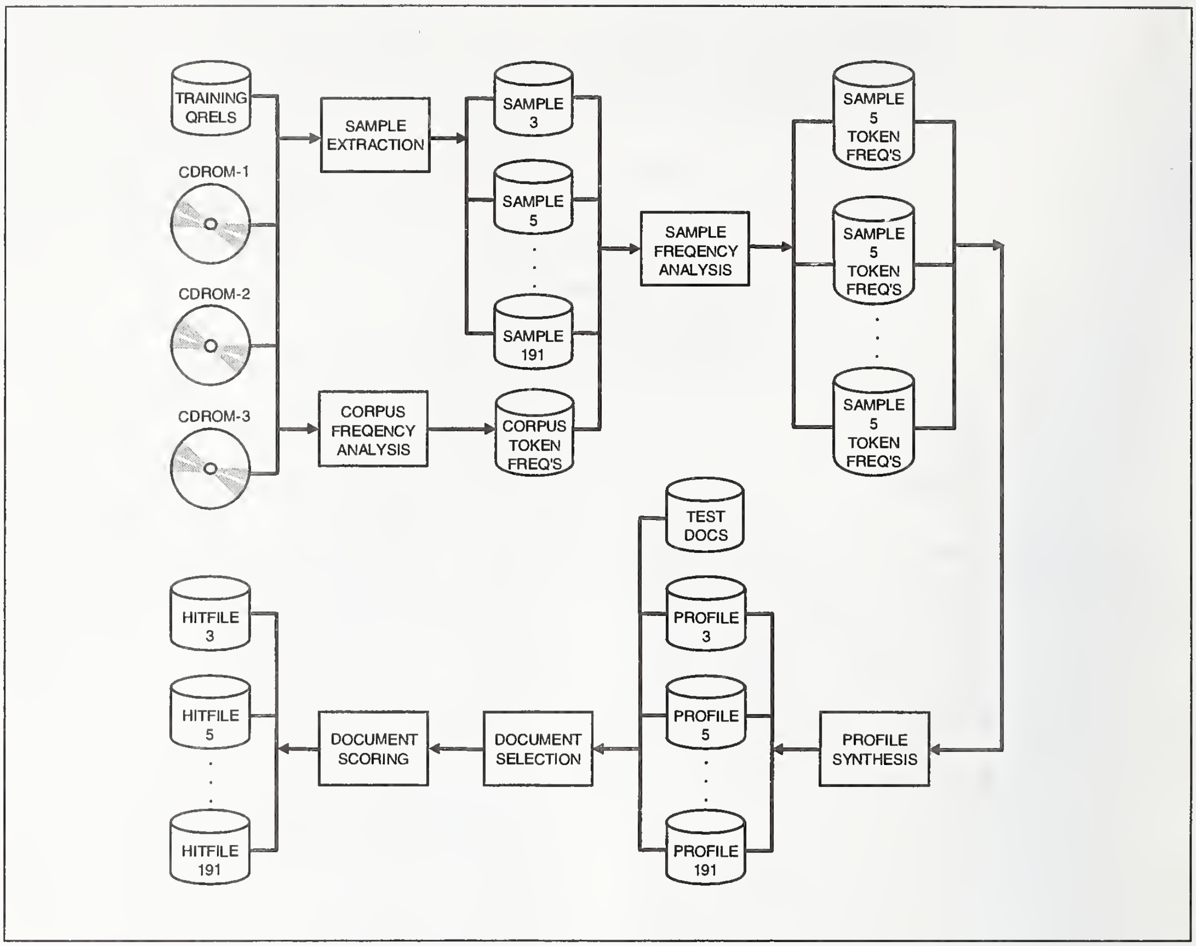


Figure 1: System Data Flow

used which reflected typical real-world resource constraints:

Processor:	Dedicated SPARCstation 10
CPU Clock:	50 MHz
RAM:	32 MB
Hard Disk:	5 GB

2. Problem Description

Routing participants in TREC-4 were provided with 1,078,000 training documents, 3.2 gigabytes on three CDROM disks. Documents were selected from the *Wall Street Journal*, *AP Newswire*, *Computer Select* disks, *Federal Register*, *San Jose Mercury News*, DOE abstracts, and US patents, and were stored with embedded SGML formatting tags.

Participants also received 50 routing topics, each a one-page description of a user's information needs, together with 50 sets of document numbers (prior TREC retrieval relevance judgments, or "QRELS") identifying the training documents which satisfied each topic.

Participants were to use the above data to generate routing profiles and to develop relevance scoring algorithms. They were then provided with 329,000 test documents (0.9 gigabytes) selected from the *Federal Register*, *Computer Select* disks, *IR Digest*, *Virtual Worlds*, and Internet newsgroups.

They were to run their routing profiles against the test documents to identify the 1,000 highest-scoring documents for each topic. They were then to sort the set of hits for each topic in descending order of relevance, and were to return the 50 sets of 1,000 hits for evaluation by TREC-4 judges.

3. System Overview

As shown in Figure 1, the prototype system consisted of six separate processes, each with its own specific purpose:

1. Corpus Frequency Analysis analyzed a large representative corpus of documents to create a database of token-frequency data.

2. Sample Extraction used the training QRELS to extract all documents relevant to each routing topic from CDROMs 1 through 3, and created a database of topic samples.

3. Sample Frequency Analysis analyzed the sample set of documents associated with each topic to determine the set of tokens that were statistically most descriptive of the sample, and assigned a weight to each token based on its observed frequencies of occurrence in the sample and in the corpus.

4. Profile Synthesis incorporated the most descriptive tokens into a LMDS profile, appended a Boolean logic statement to indicate which tokens would be used for document selection, and sent the completed profile to LMDS for activation.

5. Document Selection used the LMDS routing engine to dispatch each active profile against each of the test documents. Whenever the Boolean logic statement in a profile was satisfied by the document, LMDS sent the document to the Document Scoring process, together with the token and weighting data necessary to score the document.

6. Document Scoring calculated a document score for each profile which selected the document, based on the weights of all profile terms that were found in the document. The scoring routine then stored the document and its score in a profile-designated hitfile for TREC-4 evaluation.

The details of each of the above processes are discussed below in the context of TREC-4. Performance figures for each process are provided.

3.1. Corpus Frequency Analysis

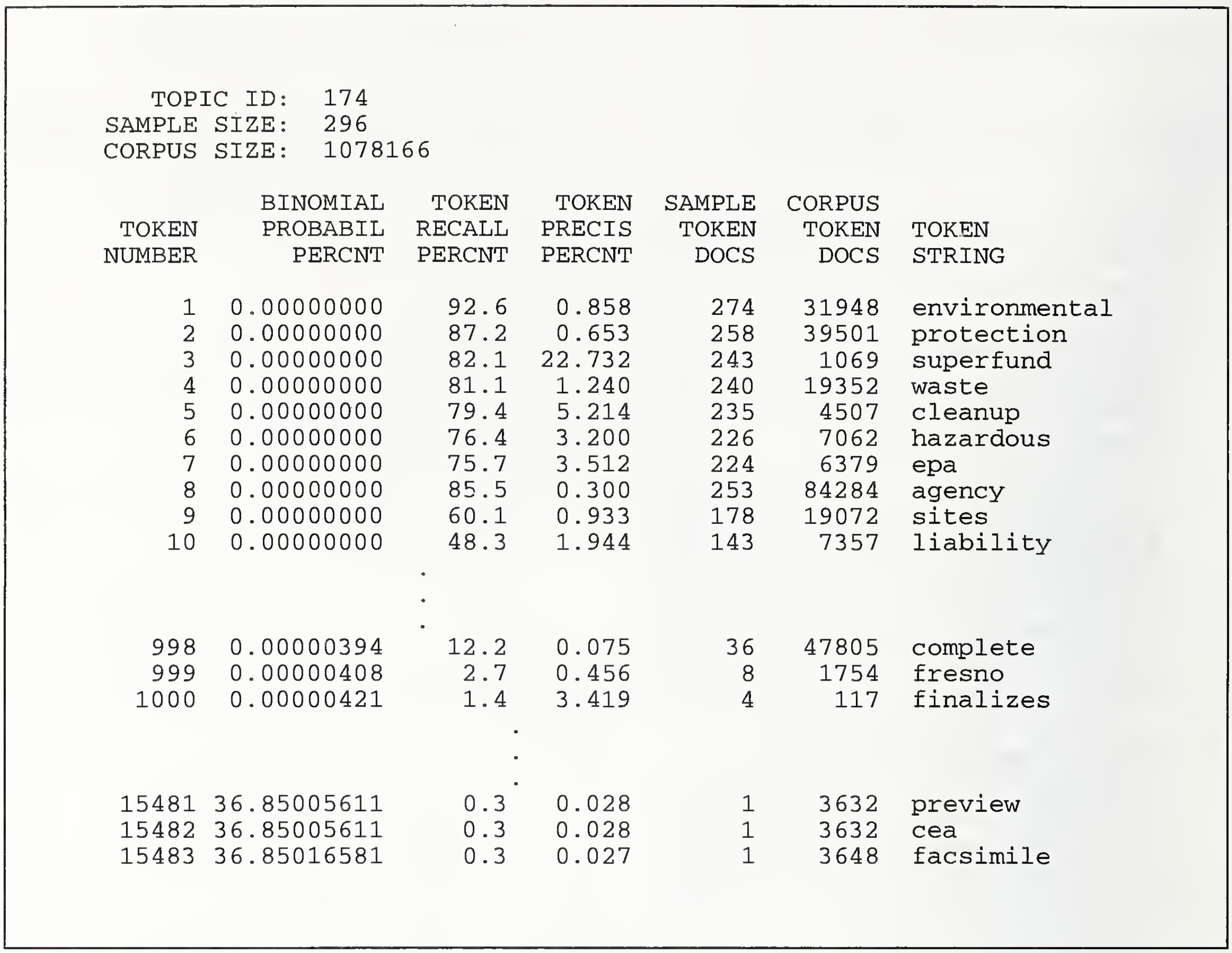
This preliminary process analyzed all training documents on CDROMs 1 through 3. (This total set of documents will be referred to hereafter as the "corpus.") For each document, the process tokenized all alphanumeric strings, and then eliminated those tokens which:

1. Were only one character long.
2. Contained one or more numeric characters.
3. Were in the stopword list, a slightly augmented version of the list used by SMART [3].
4. Served as SGML tags.
5. Occurred <= twice on each CDROM.
6. Were in an area of the document deemed non-searchable by TREC-4 rules.

The process next counted the number of documents in which each token occurred, then

```
CORPUS SIZE: 1078166

CORPUS
 TOKEN   TOKEN
  DOCS   STRING

   712   aa
   201   aaa
     2   aaaa
     2   aaac
          .
          .
          .
     9   superfonts
     4   superforms
     4   superfos
    56   superframe
     5   superfunction
  1069   superfund
     4   superfundlist
     7   superfused
     6   superfusion
    18   superguage
    10   supergen
          .
          .
          .
   114   zz
     6   zzz
    88   zzzz
     7   zzzzbest
```

Figure 2: Corpus Analysis Results



```
   TOPIC ID:   174
SAMPLE SIZE:   296
CORPUS SIZE:   1078166

             BINOMIAL   TOKEN    TOKEN   SAMPLE  CORPUS
  TOKEN      PROBABIL  RECALL   PRECIS    TOKEN   TOKEN  TOKEN
NUMBER         PERCNT  PERCNT   PERCNT     DOCS    DOCS  STRING

      1    0.00000000    92.6    0.858      274   31948  environmental
      2    0.00000000    87.2    0.653      258   39501  protection
      3    0.00000000    82.1   22.732      243    1069  superfund
      4    0.00000000    81.1    1.240      240   19352  waste
      5    0.00000000    79.4    5.214      235    4507  cleanup
      6    0.00000000    76.4    3.200      226    7062  hazardous
      7    0.00000000    75.7    3.512      224    6379  epa
      8    0.00000000    85.5    0.300      253   84284  agency
      9    0.00000000    60.1    0.933      178   19072  sites
     10    0.00000000    48.3    1.944      143    7357  liability
                    .
                    .
                    .
    998    0.00000394    12.2    0.075       36   47805  complete
    999    0.00000408     2.7    0.456        8    1754  fresno
   1000    0.00000421     1.4    3.419        4     117  finalizes
                            .
                            .
                            .
  15481   36.85005611     0.3    0.028        1    3632  preview
  15482   36.85005611     0.3    0.028        1    3632  cea
  15483   36.85016581     0.3    0.027        1    3648  facsimile
```

Figure 3: Sample Analysis Results for Topic 174

sorted the tokens alphabetically and stored them in a file with their associated document counts. A portion of this file is shown in Figure 2. The following figures summarize the performance of the process:

1,078,000	Documents analyzed
580	Stopwords used
249,000	Tokens identified
0.03	Wall-clock seconds/document, average

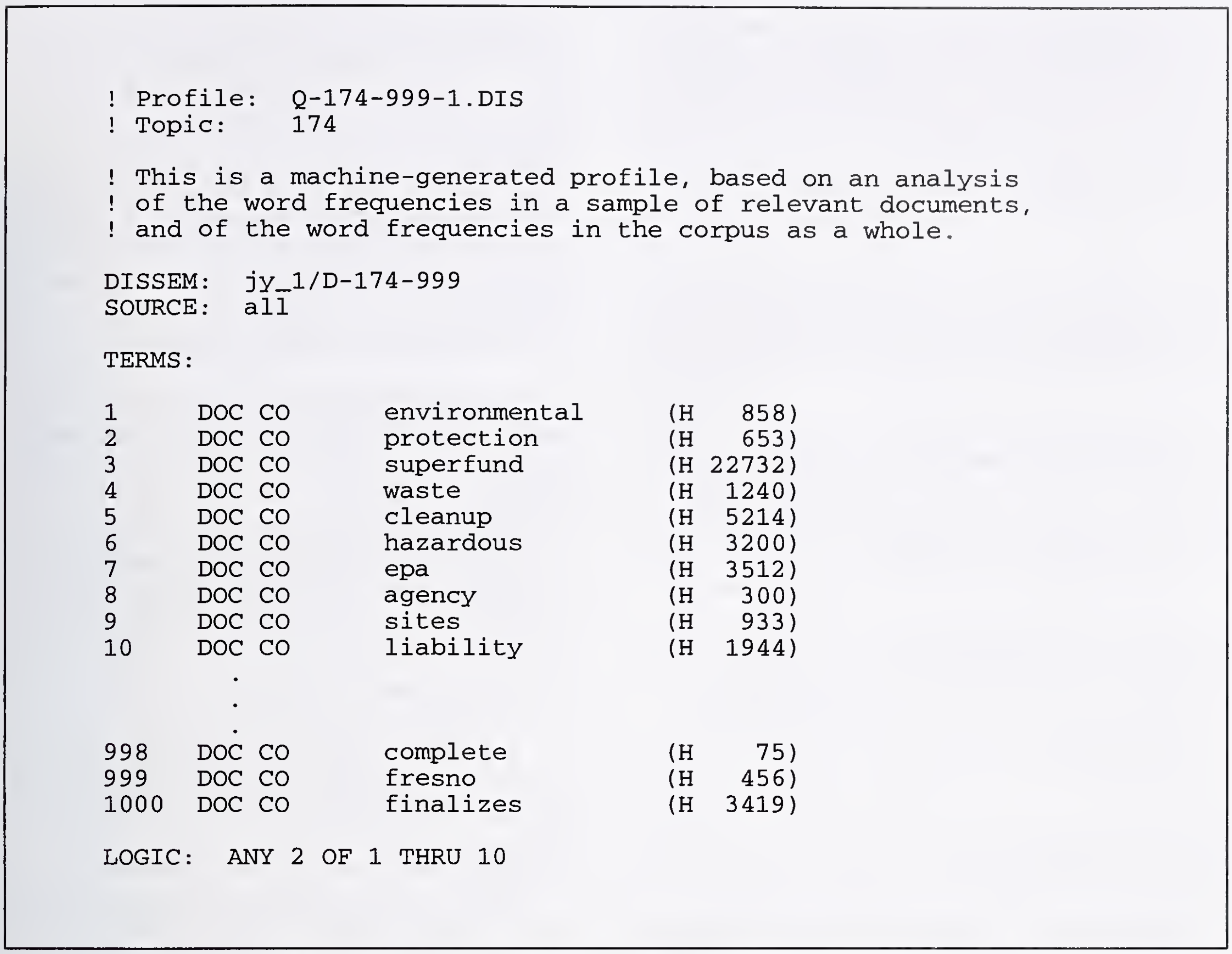
3.2. Sample Extraction

This preliminary process used the training QRELS to extract all relevant documents for each topic from the corpus, and to store these sets of documents into 50 separate directories. Each directory thus contained the sample for a given topic. No figures were kept on the time required to perform this extraction, but sample sizes varied from a low of 28 documents to a high of 803 documents.

3.3. Sample Analysis

This process was responsible for producing the sample statistics necessary for automatic profile generation. For each topic sample, the process first tokenized each document in the sample, using the same tokenizing rules as Corpus Frequency Analysis.

The process then counted the number of sample documents in which each token occurred. The process then combined the sample count for each token with the corpus count for that token to



```
! Profile:  Q-174-999-1.DIS
! Topic:    174

! This is a machine-generated profile, based on an analysis
! of the word frequencies in a sample of relevant documents,
! and of the word frequencies in the corpus as a whole.

DISSEM:  jy_1/D-174-999
SOURCE:  all

TERMS:

1     DOC CO     environmental     (H   858)
2     DOC CO     protection        (H   653)
3     DOC CO     superfund         (H 22732)
4     DOC CO     waste             (H  1240)
5     DOC CO     cleanup           (H  5214)
6     DOC CO     hazardous         (H  3200)
7     DOC CO     epa               (H  3512)
8     DOC CO     agency            (H   300)
9     DOC CO     sites             (H   933)
10    DOC CO     liability         (H  1944)
        .
        .
        .
998   DOC CO     complete          (H    75)
999   DOC CO     fresno            (H   456)
1000  DOC CO     finalizes         (H  3419)

LOGIC:  ANY 2 OF 1 THRU 10
```

Figure 4: Automatically Generated Profile for Topic 174

calculate the binomial probability distribution *P(r)* for the token, as shown in the following formula:

$$P(r) = \frac{n!}{r!(n-r)!}\left(\frac{p}{q}\right)^{r}\left(1-\frac{p}{q}\right)^{n-r}$$

where:

n = documents in sample
r = sample documents containing token
p = corpus documents containing token
q = documents in corpus

Calculated in this way, the value *P(r)* can be used as a measure of how "descriptive" each token is with regard to a given sample of documents, with lower values indicating greater descriptive power.

The process next calculated the weight *w* for each token, as shown in the following formula:

$$w = \frac{r}{p}$$

The process then created a table containing each unique token in the topic sample, together with its associated values for *P(r)* and *w*. Finally, the process sorted this table on *P(r)*, the measure of descriptiveness.

A portion of one such table is shown is Figure 3. Its rows contain the statistics for each unique token in the topic sample. Its columns contain the information described below:

1. Token Number is a one-up token identification number.

2. Binomial Probability Percent is the value *P(r)* expressed as a percent -- the probability that this token has occurred in this sample as often as it has, purely by chance.

3. Token Recall Percent is the value:

$$\frac{r}{n}$$

expressed as a percent -- the probability that this token will occur in a relevant document.

4. Token Precision Percent is the value *w* expressed as a percent -- the probability a document will be relevant if it contains this token.

5. Sample Token Documents is the value *r* -- the total sample documents containing the token.

6. Corpus Token Documents is the value *p* -- the total corpus documents containing the token.

7. Token String is the first 14 characters of the token.

3.4. Profile Synthesis

To create the actual LMDS profile, this process concatenated the top 1000 tokens with their respective weights from the sorted table, and placed them in a file. It then appended a Boolean logic statement indicating that a document was to be selected if it contained any two of the top ten most descriptive tokens. (Once a document was selected, however, the scoring algorithm would assign a document score based on the weights of all profile tokens that appeared in that document.) The process then sent the new profile to LMDS to be activated. A portion of one such profile is shown in Figure 4.

The following figures summarize the combined performance of the Sample Analysis and Profile Synthesis processes:

50	Profiles generated
5.4	Wall-clock minutes/profile, average
0.8	Wall-clock seconds/document, average

3.5. Document Selection

Document selection was performed with the LMDS 2.1.4 routing engine, which enabled each profile to perform a free-text scan of each test document.

Whenever the Boolean logic statement in any profile was satisfied by a document, LMDS sent the

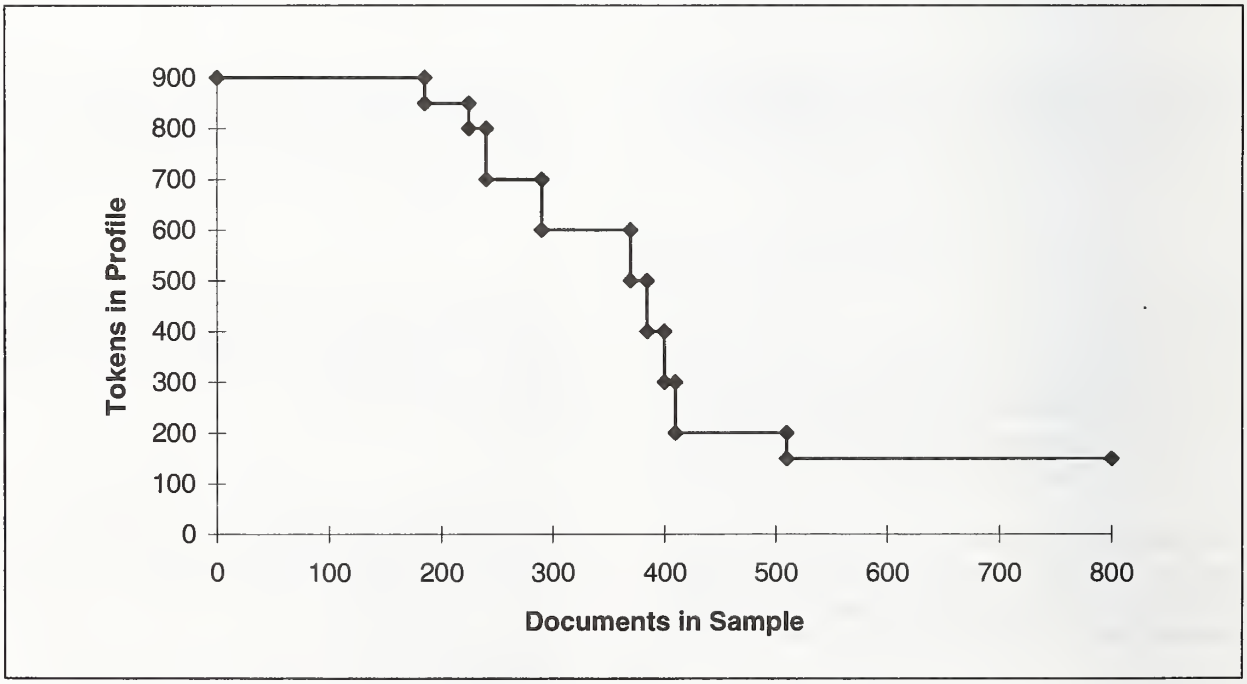


Figure 5: Observed Optimum Relationship of Sample Size to Profile Size

document to the Document Scoring process, together with the token and weighting information needed by that process to score the document.

The LMDS program product itself was not modified in any way for TREC-4. All prototype functionality was added via externally generated LMDS profiles and via the LMDS API.

3.6. Document Scoring

For each document and set of profile tokens and weights received from LMDS, this process first tokenized the document, using the same tokenizing rules as Corpus Frequency Analysis. The process next divided the document into a set of "passages," where each passage consisted of 100 contiguous tokens, and each new passage began 50 tokens after the beginning of the previous passage. The process then calculated a set of intermediate scores, consisting of:

1. An intermediate score for each passage:

$$s_p = (1-(1-w_1)(1-w_2)...(1-w_n))$$

2. An intermediate score for the entire document:

$$s_d = (1-(1-w_1)(1-w_2)...(1-w_n))\frac{u}{t}$$

where:

n = unique profile tokens in passage or document
w_i = weight of each unique profile token in passage or document
u = unique profile tokens in document
t = unique tokens in document

The passage formula calculated the probability of relevance to a topic, given the presence of a particular subset of the topic's most descriptive tokens. The document formula did the same, except that it multiplied by the ratio of *u/t* to correct for the tendency of larger documents to achieve higher scores simply by having more tokens.

Although each profile contained 1,000 tokens, it was discovered during TREC-3 testing that as the sample size increased, the accuracy of the profile usually decreased [2]. Further testing suggested that the optimum number of profile tokens indeed was inversely proportional to the size of the sample. Accordingly, the Document Scoring process was modified to calculate each intermediate score with only the top *x* tokens in the profile, where *x* was the value of a step function on the sample size, as shown in Figure 5.

When all intermediate scores had been calculated, the process then combined the scores in two different ways to produce two final document scores, one for each TREC-4 run (losPA2 and losPA3):

1. For losPA2, thc process used the highest passage score for each document to determine the top 1000 documents for each topic, then assigned the intermediate document score as the final document score.

2. For losPA3, the process multiplied the intermediate document score by two, then added the highest passage score to produce the final score.

For each run, the process then stored the document number and its score into the hitfile designated by the profile. When all test documents had been processed in this fashion, the 50 hitfiles were sorted on document score, and the top 1,000 entries in each hitfile were sent to the TREC-4 judges.

The following figures summarize the combined performance of the Document Selection and Document Scoring processes:

329,000	Documents processed
0.45	Wall-clock seconds/document, average
27,000	Documents scored per topic, average

4. TREC-4 Results

The detailed results for runs losPA2 and losPA3 are provided in Figure 6. Run losPA3 achieved better results, and its recall-precision scores are summarized in the graph in Figure 7.

Except for the first section of Figure 6, all scores are averages of the corresponding scores for each of the 50 topics. While most sections of Figure 6 are self-explanatory, several concepts may require additional explanation:

```
Run ID:          losPA2   losPA3

Total docs over all topics:
  Retrieved:    50000    50000
  Relevant:      6576     6576
  Ret AND Rel:   4993     4957

Interpolated recall-precision:
     At 0.00:   0.7848   0.7553
     At 0.10:   0.5105   0.5174
     At 0.20:   0.4436   0.4505
     At 0.30:   0.3738   0.3796
     At 0.40:   0.3262   0.3234
     At 0.50:   0.2817   0.2820
     At 0.60:   0.2293   0.2354
     At 0.70:   0.1781   0.1874
     At 0.80:   0.1303   0.1293
     At 0.90:   0.0598   0.0668
     At 1.00:   0.0114   0.0122

Non-interpolated precision
over all relevant documents:
                0.2793   0.2834

Precision at:
      5 docs:   0.5160   0.5160
     10 docs:   0.4940   0.4960
     15 docs:   0.4840   0.4840
     20 docs:   0.4700   0.4670
     30 docs:   0.4433   0.4527
    100 docs:   0.3578   0.3550
    200 docs:   0.2757   0.2828
    500 docs:   0.1651   0.1651
   1000 docs:   0.0999   0.0991

R-Precision:
                0.3108   0.3161
```

Figure 6: Routing Results: losPA2 and losPA3

1. Interpolated precision is the maximum precision over a range of recall points. Thus, the interpolated precision at recall 0.10 (i.e., after 10% of relevant documents have been retrieved for a query) is the maximum precision at all recall points >= 0.10.

2. Precision at X docs is the precision after X documents have been retrieved, whether or not the documents are relevant.

3. R-Precision is the precision after R documents have been retrieved, where R is the number of relevant documents possible for a topic. Thus, if a topic has 50 possible relevant documents, precision is measured for that topic only at 50 documents.

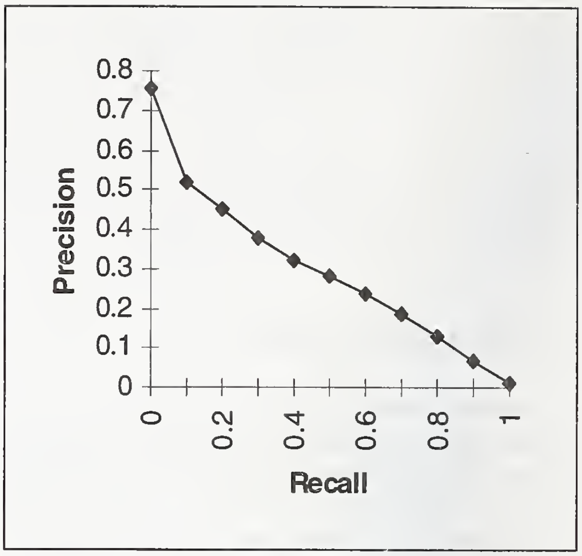


Figure 7: Recall-Precision Curve: losPA3

5. Analysis

To judge the effect of the improvements incorporated into our TREC-4 prototype, we performed a benchmark after the conference, using our TREC-3 algorithm [2] with the TREC-4 training QRELS, test documents, and scoring software. The benchmark revealed modest improvements in both recall (2.1%) and precision (5.0%) over TREC-3, and its results are summarized in the first three rows of the table in Figure 8.

The fourth row of this table contains the results of an experimental run, also performed since the conference, using two-token phrases for search terms. This phrase-based approach achieved substantial improvements both in recall (9.1%) and precision (16.9%) over the TREC-3 approach, and is discussed in more detail in the Appendix.

The columns in Figure 8 contain the following data:

1. Algorithm identifies the approach which produced the results. "TREC-3" here refers to the benchmark using the Logicon TREC-3 algorithm with the TREC-4 training QRELS and test documents. "Phrases" here indicates the experimental algorithm described in the Appendix.

2. Recall is the total number of relevant documents retrieved divided by the total number of relevant documents.

Algorithm	Recall	Average Precision	Recall Improvement Over TREC-3 Baseline	Precision Improvement Over TREC-3 Baseline
TREC-3*	73.81%	26.99%		
losPA2	75.93%	27.93%	2.87%	3.48%
losPA3	75.38%	28.34%	2.13%	5.00%
Phrases*	80.58%	31.55%	9.17%	16.90%
*Unofficial results, using TREC-4 QRELS, documents, and scoring software				

Figure 8: Comparison of Logicon Results

3. Average Precision is the non-interpolated average precision over all relevant documents.

4. Recall Improvement Over TREC-3 Baseline is the percentage difference between Recall and "TREC-3" Recall.

5. Precision Improvement Over TREC-3 Baseline is the percentage difference between Average Precision and "TREC-3" Average Recall.

6. Summary

The prototype described in this paper is a hybrid routing system, successfully coupling the LMDS Boolean search engine with a probabilistic scoring algorithm, and using the speed of LMDS to quickly reduce the set of documents that must be evaluated by the more computationally intensive scoring algorithm. As implemented, the prototype is simple, compact and fast, and requires no special hardware.

Processing logs show that, for any given topic, the Boolean-based Document Selection process sent an average of only 8.2% of the test documents to the Document Scoring process (27,000 of 329,000), yet this small set still contained an average of 96% of the documents relevant to the topic. Moreover, the average wall-clock time required to completely process each document against all topics was only 0.45 second per document.

Passage-based routing has improved our scores compared to TREC-3, and phrase-based experiments promise substantial improvements still to come.

7. References

[1] J. Yochum. A High-Speed Text Scanning Algorithm Utilizing Least Frequent Trigraphs. In *Proceedings of the IEEE International Symposium on New Directions in Computing*, pages 114-121, Trondheim, Norway, 1985.

[2] J. Yochum. Research in Automatic Profile Generation and Relevance Ranking with LMDS. In D. Harman, editor, *Overview of the Third Text REtrieval Conference (TREC-3)*, pages 289-297. National Institute of Standards and Technology Special Publication 500-225, 1995.

[3] C. Buckley. *Implementation of the SMART Information Retrieval System*, Technical Report TR 85-686, Computer Science Department, Cornell University, Ithaca, NY, 1985.

APPENDIX

An Experiment with Phrase-Based Routing

A1. Introduction

This appendix describes a set of extensions to the TREC-4 algorithm, made since the conference, to incorporate the use of two-token phrases as search terms.

The primary effort required by this experiment involved the development of a program to identify the phrases in a document. Minor modifications were then made to several of the processes of the TREC-4 prototype to utilize this program. The program and the modifications are described below.

A2. Phrase Identification Program

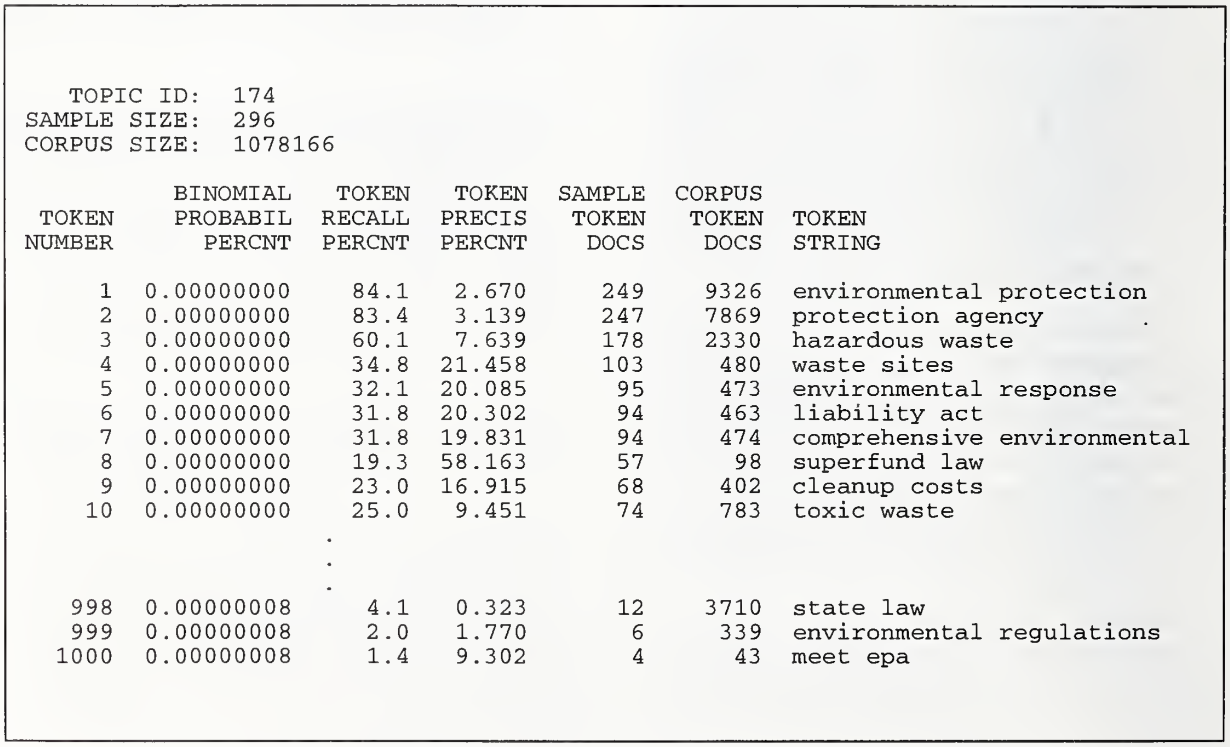
To locate the phrases within a document, this program first translated each non-alphanumeric character in the document to a blank, leaving a set of alphanumeric tokens separated by strings of one or more blanks. The program then translated to a blank any token which:

1. Was only one character long.
2. Contained one or more numeric characters.
3. Was in the TREC-4 algorithm stopword list.
4. Served as an SGML tag.
5. Was in an area of the document deemed non-searchable by TREC-4 rules.

When this process was complete, the program treated as a phrase any two tokens separated by only a single blank.

A3. System Overview

The phrase-based approach used the same six processes as the TREC-4 algorithm. Accordingly, each process is described below only as it differed from its associated TREC-4 process.



```
   TOPIC ID:   174
SAMPLE SIZE:   296
CORPUS SIZE:   1078166

           BINOMIAL   TOKEN    TOKEN  SAMPLE  CORPUS
 TOKEN     PROBABIL  RECALL   PRECIS   TOKEN   TOKEN  TOKEN
NUMBER       PERCNT  PERCNT   PERCNT    DOCS    DOCS  STRING

     1  0.00000000    84.1    2.670     249    9326  environmental protection
     2  0.00000000    83.4    3.139     247    7869  protection agency
     3  0.00000000    60.1    7.639     178    2330  hazardous waste
     4  0.00000000    34.8   21.458     103     480  waste sites
     5  0.00000000    32.1   20.085      95     473  environmental response
     6  0.00000000    31.8   20.302      94     463  liability act
     7  0.00000000    31.8   19.831      94     474  comprehensive environmental
     8  0.00000000    19.3   58.163      57      98  superfund law
     9  0.00000000    23.0   16.915      68     402  cleanup costs
    10  0.00000000    25.0    9.451      74     783  toxic waste
                        .
                        .
                        .
   998  0.00000008     4.1    0.323      12    3710  state law
   999  0.00000008     2.0    1.770       6     339  environmental regulations
  1000  0.00000008     1.4    9.302       4      43  meet epa
```

Figure A1: Sample Analysis Results (Phrases) for Topic 174

1. Corpus Frequency Analysis was augmented to create a database of phrase-frequency data in addition to its database of token-frequency data.

2. Sample Extraction was not modified.

3. Sample Frequency Analysis was augmented to also determine the set of phrases that were statistically most descriptive of the sample, and to assign a weight to each phrase based on its observed frequencies of occurrence in the sample and in the corpus. Sample phrase-related output from this process is shown in Figure A1.

4. Profile Synthesis was modified to merge both the most descriptive tokens and the most descriptive phrases into a LMDS profile, and to append a Boolean logic statement indicating which tokens and phrases would be used for document selection. A sample profile produced by this process is shown in Figure A2.

5. Document Selection was not modified.

6. Document Scoring was modified to calculate only the intermediate score for each passage:

$$s_p = (1-(1-w_1)(1-w_2)...(1-w_n))$$

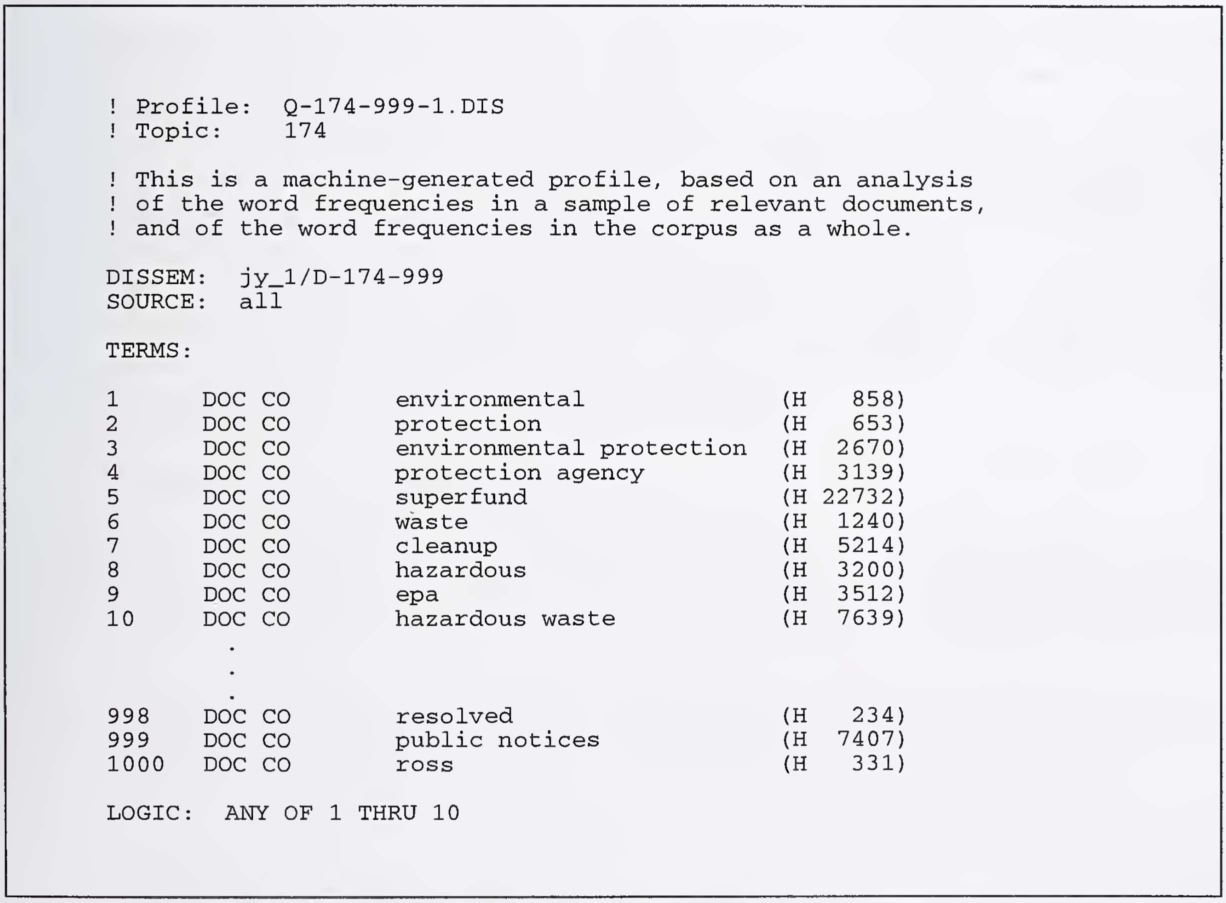
where:

w_i = *weight of each unique profile token or phrase in the passage*

The process then assigned the highest passage score as the final document score.

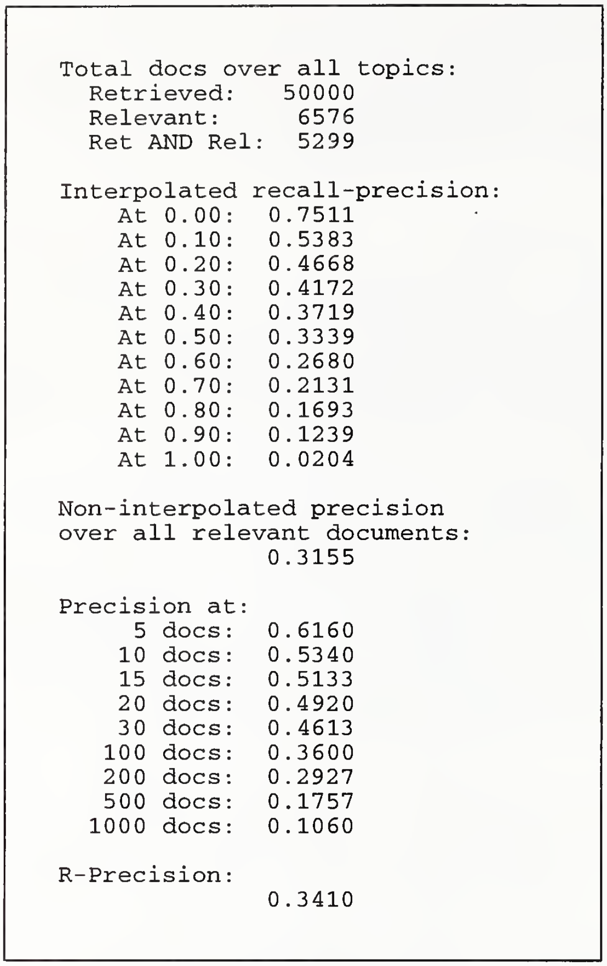
A4. Phrase-Based Results

To judge the effectiveness of the phrase-based modifications described above, we ran a benchmark using the TREC-4 training QRELS, test documents, and scoring software. The detailed results of this run are provided in Figure A3.



```
! Profile:  Q-174-999-1.DIS
! Topic:    174

! This is a machine-generated profile, based on an analysis
! of the word frequencies in a sample of relevant documents,
! and of the word frequencies in the corpus as a whole.

DISSEM:  jy_1/D-174-999
SOURCE:  all

TERMS:

1      DOC CO      environmental              (H   858)
2      DOC CO      protection                 (H   653)
3      DOC CO      environmental protection   (H  2670)
4      DOC CO      protection agency          (H  3139)
5      DOC CO      superfund                  (H 22732)
6      DOC CO      waste                      (H  1240)
7      DOC CO      cleanup                    (H  5214)
8      DOC CO      hazardous                  (H  3200)
9      DOC CO      epa                        (H  3512)
10     DOC CO      hazardous waste            (H  7639)
         .
         .
         .
998    DOC CO      resolved                   (H   234)
999    DOC CO      public notices             (H  7407)
1000   DOC CO      ross                       (H   331)

LOGIC:  ANY OF 1 THRU 10
```

Figure A2: Automatically Generated Profile (Phrases) for Topic 174



```
Total docs over all topics:
  Retrieved:   50000
  Relevant:     6576
  Ret AND Rel:  5299

Interpolated recall-precision:
    At 0.00:  0.7511
    At 0.10:  0.5383
    At 0.20:  0.4668
    At 0.30:  0.4172
    At 0.40:  0.3719
    At 0.50:  0.3339
    At 0.60:  0.2680
    At 0.70:  0.2131
    At 0.80:  0.1693
    At 0.90:  0.1239
    At 1.00:  0.0204

Non-interpolated precision
over all relevant documents:
              0.3155

Precision at:
     5 docs:  0.6160
    10 docs:  0.5340
    15 docs:  0.5133
    20 docs:  0.4920
    30 docs:  0.4613
   100 docs:  0.3600
   200 docs:  0.2927
   500 docs:  0.1757
  1000 docs:  0.1060

R-Precision:
              0.3410
```

Figure A3: Routing Results: Phrases

A5. Summary

As was noted above in Section 5 and was shown in Figure 8, the phrase-based approach achieved significant improvements both in recall (9.1%) and in precision (16.9%) over the baseline TREC-3 algorithm.

These results are highly encouraging, and make it clear that phrases will be an important addition to future versions of the algorithm.

The TREC-4 Filtering Track

David D. Lewis
AT&T Research
600 Mountain Avenue
Murray Hill, NJ 07974
lewis@research.att.com

Abstract

The TREC-4 filtering track was an experiment in the evaluation of binary text classification systems. In contrast to ranking systems, binary text classification systems may need to produce result sets of any size, requiring that sampling be used to estimate their effectiveness. We present an effectiveness measure based on utility, and two sampling strategies (pooling and stratified sampling) for estimating utility of submitted sets. An evaluation of four sites was successfully carried out using this approach.

1 Introduction

The goal of the TREC-4 filtering track was to develop methods for evaluating binary text classification systems, and try out those methods on real data. A secondary goal was to give TREC participants their first chance to evaluate approaches to binary text classification.

This paper begins by defining binary text classification and presenting some applications of it. We then discuss a particular binary text classification task, filtering, used in TREC-4. The effectiveness of filtering submissions was evaluated using utility as a measure. The several roles that this effectiveness measure played in the evaluation are described. The large size of the TREC-4 test data set meant that relevance judgments were of necessity incomplete and effectiveness could only be estimated. We describe in detail two approaches that were tested for estimating the utility of filtering submissions. We then briefly discuss the results of the TREC-4 filtering track, with an emphasis on what was learned about the evaluation methods.

2 Binary Text Classification

By binary text classification systems, we mean information retrieval (IR) systems that decide for each document processed whether the document should be accepted or rejected [6]. What it means to be accepted varies between systems. Some applications that make use of binary text classification are:

- A company provides an SDI (selective dissemination of information) service which filters newswire feeds. Relevant articles are faxed each morning to clients.

- A text categorization system assigns controlled vocabulary categories to incoming documents as they are stored in a text database.

- An "agent" program monitors low content text streams (e.g. Usenet newsgroups) and alerts a user when a relevant message appears.

Note that there is no notion of a user choosing how far to go down a ranking in these systems. The system makes Yes/No decisions about documents, and the user only sees the results of those decisions. This affects the kind of evaluation appropriate for the system, as discussed in the next section.

3 Evaluation for Binary Text Classification

We have discussed evaluation for binary classification at length elsewhere [5, 6], and so here will concentrate on how it differs from the evaluation of ranked retrieval in the main TREC-4 tasks.

Effectiveness measures for ranked retrieval typically have two components. The first is a *cutoff*: a specification of how to divide the ranking into a top part and a bottom part. The top part is considered to be the set of documents retrieved by the system. The second component of the overall effectiveness measure is a *set-based effectiveness measure* which is applied to the retrieved set.

Some examples of these two-component measures are:

- Precision at 0.10 recall : Here the cutoff is the highest point in the ranking above which at least 10% of the relevant documents in the test set occur. The set-based effectiveness measure is precision, the proportion of documents in the retrieved set which are relevant.

- Recall at 0.001 fallout : Similar to the above, but the cutoff is based on fallout and the set-based effectiveness measure is recall.

- Precision at 20 documents : Here the cutoff is based on a fixed position in the ranking (20 documents down from the top).

- R-precision : This is precision at R documents, where R is the number of relevant documents in the collection.

More complex measures, such as the average of precision over multiple recall cutoffs are also used. A wide variety of such measures with different cutoffs and different set-based effectiveness measures have been applied to rankings in the TREC evaluations. This might seem to make it difficult for a site to decide how rank documents, since there is more than one measure to optimize. In truth, all these measures can be optimized simultaneously by the simple and obvious strategy of ranking documents by how likely they are to be relevant. Doing so will result in an optimal score under essentially any reasonable measure of ranking effectivenes, a property which has been formalized as the Probability Ranking Principle [6, 9].

In contrast, binary classification systems make the separation into accepted and rejected documents themselves, rather than leaving this up to the effectiveness measure used in evaluation. The binary classification system must choose what separation to make in order to optimize the effectiveness measure used. Doing so optimally means that the effectiveness measure must be known in advance. Since binary classification systems do not rank the accepted set, the effectiveness measure should be a set-based one. Since the size of the submitted set is under the control of the system, the effectiveness measure used in evaluation must be able to assign an effectiveness to a set of any size, including the empty set.

It is still desirable in the filtering context to test the ability of systems to satisfy varying user preferences (e.g. high recall vs. high precision), but this should not be done by submitting a single ranking and letting the evaluation program pick cutoffs. Instead, a family of effectiveness measures can be used to capture different user preferences. For each measure used, the filtering system produces a separate set of documents appropriate to that measure, and that same measure is used to evaluate the set of documents.

4 Filtering: A Binary Classification Task for TREC

The mainline tasks for TREC-1 through TREC-4, routing and ad hoc retrieval, require participants to submit ranked lists of documents, which are then evaluated using ranking-oriented effectiveness measures. The number of documents submitted is defined in advance, so the ability of systems to pick the number of documents to submit is not tested.

The TREC-4 filtering track addresses this limitation. This section describes the rationale for the evaluation, the evaluation's structure, and the effectiveness measures used.

4.1 Why Filtering?

The main motivation for the filtering track is the increasing number of IR applications requiring binary text classification (see Section 2). The track should help developers of these applications learn about relevant techniques from the research community, and let researchers compare and evaluate their approaches.

A second motivation is that the demands of the filtering task may encourage the development of IR methods with other desirable properties. For instance, accurately estimating the probability of relevance of documents is useful not only in filtering [6], but also for self-monitoring of effectiveness [6] and selection of training data [7].

Finally, we hope that a binary classification task will attract a broader range of researchers and approaches to TREC. The requirement that TREC results be ranked makes it awkward for approaches that are not ranking-oriented to be tried [4, 10]. These approaches include boolean querying by human experts, as well as the use of binary classifiers produced by machine learning techniques.

4.2 Structure

The structure of the filtering evaluation was as follows. Systems were given 150 descriptions of user needs. Each description was made up of a *topic* describing the kind of information sought, and a *set-based effectiveness measure* capturing a hypothetical user's tolerance for different kinds of mistakes. The topics were the same 50 topics used in the main TREC-4 routing evaluation. Three effectiveness measures (see Section 4.4) were combined with each of the 50 topics, to yield the 150 user needs.

For each user need, the system had to make use of the topic and the effectiveness measure to decide whether to accept or reject each test document. The same test documents as the main routing evaluation were used. The submitted set (i.e. the accepted documents) for the user need was then evaluated using the effectiveness measure for that user need. (Actually only a sample from the submitted set was used—see Section 5.)

4.3 Integration with Routing

The TREC-4 filtering and routing evaluations used the same topics and test documents. (See Harman's discussion of these elsewhere in the proceedings.) The training data for the topics (i.e. documents judged with respect to the topics in previous TRECs) were also the same in both evaluations.

The evaluations were also similar in that 100 documents from each site's results for a topic went into the pool for judging. In the case of routing, the judged documents were the top 100 from a single ranked list of 1000 documents submitted for the topic. In the case of filtering, the 100 documents were a stratified sample (Section 5.2.1) from the union of the three unranked sets of documents submitted for that topic. Filtering and routing documents were mixed together and treated identically for judging. The expense of running the filtering track was thereby reduced, since some filtering and routing submissions overlapped.

4.4 The Utility Measures

The family of effectiveness measures used in the filtering track were based on assigning a numeric value or *utility* to each retrieved document [2, 6]. Retrieved relevant documents received a positive utility, and retrieved nonrelevant documents received a negative utility. The total utility measures of a submitted set was:

$$u_i = u_{ai}A_i + u_{bi}B_i$$

where A_i is the number of relevant documents in the submitted set for run R_i and B_i is the number of nonrelevant documents in the submitted set for run R_i. For each run R_i we assume that u_{ai} is the value the user places on receiving a relevant document, while u_{bi} is the value of receiving a nonrelevant document.

Different values for u_{ai} and u_{bi} define different effectiveness measures in the family. The three utility measures used in the filtering evaluation were:

	Parameter	Effectiveness
Run	Values	Measure
R_1	$u_{a1} = 1, u_{b1} = -3$	$u_1 = A_1 - 3B_1$
R_2	$u_{a2} = 1, u_{b2} = -1$	$u_2 = A_2 - B_2$
R_3	$u_{a3} = 3, u_{b3} = -1$	$u_3 = 3A_3 - B_3$

We might imagine run R_1 corresponding to a user is willing to pay 1 dollar (or pick your favorite currency) for each relevant document, and 3 dollars to avoid having to read a nonrelevant document. Therefore, run R_1 requires the filtering system to act in a conservative or high precision fashion. In constrast, run R_3 encourages systems to instead emphasize high recall, while run R_2 is in between.

Unlike recall, but like precision, our utility measures take into account only accepted documents. (It is possible to define utility measures to take into account rejected documents as well [6].) Unlike both recall and precision, the total utility is not normalized to lie between 0 and 1. Indeed, there is no way to know what the maximum achievable is for any given topic, since it depends on the (unknown) total number of relevant documents in the test set. The goal of systems is simply to achieve the highest utility they can.

5 Estimating Total Utility from a Sample

Computing the exact total utility for a submitted run requires knowing the value of A_i and B_i for that run. This would require assessing the relevance of every document submitted for that run. Because submitted sets can be of any size, this might require too much work from the relevance assessors. For that reason, total utility for filtering runs was estimated using *samples* of the submitted documents.

Two different sampling and estimation methods were tried in the TREC-4 filtering track, as described below.

5.1 Pooling

The first approach to sampling was the usual TREC pooling strategy [3]. This approach assumes that some known pool of documents contains all the relevant documents in the test set. The pool for the TREC-4 filtering task consisted of all documents judged for the topic in the main routing task, plus all documents judged for the topic for the filtering task, as chosen by the stratified sampling scheme of Section 5.2.1.) Under the pooling assumption an estimate $\hat{u}_i$ of the total utility u_i is easily computed.

The total utility computed in this fashion is only an estimate, because the pooling assumption may be wrong. There may be submitted documents that are relevant but were not judged for this topic. An advantage of the pooled estimate, however, is that the same sample is used for all sites, enabling that sample to be large. The use of the same sample for all sites also eliminates a possible source of variation between sites. A disadvantage is that the sample is not a random sample, meaning that it is difficult to tell how accurate the estimated utilties are (see Section 7.1). There is also the danger that pooled sampling penalizes sites which submit atypical yet relevant documents.

5.2 Random Sampling

To see how random sampling could be used to estimate the total utility it is useful to rewrite the formula for total utility as:

$$\begin{aligned} u_i &= u_{ai} \times A_i + u_{bi} \times B_i \\ &= u_{ai} \times p_i N_i + u_{bi} \times (1 - p_i) N_i \\ &= ((u_{ai} - u_{bi}) p_i + u_{bi}) N_i \qquad (1) \end{aligned}$$

where $N_i = A_i + B_i$ is the total number of documents submitted for the run, and $p_i = A_i / N_i$ is the proportion of documents submitted which are relevant (i.e. the precision of the submission). Rewriting the utility measures used in the filtering evaluation in this way gives:

$$\begin{aligned} u_1 &= (4p_1 - 3) N_1 \qquad (2) \\ u_2 &= (2p_2 - 1) N_2 \qquad (3) \\ u_3 &= (4p_3 - 1) N_3 \qquad (4) \\ & \qquad (5) \end{aligned}$$

Therefore, if we can produce an estimate $\hat{p}_i$ of the proportion of relevant documents in a submitted set, we can turn that into an estimate, $\hat{u}_i$, of the utility of the submitted set:

$$\hat{u}_i = ((u_{ai} - u_{bi}) \hat{p}_i + u_{bi}) N_i \qquad (6)$$

One approach to estimating the proportion would be to take a random sample from the submitted set for a topic/run pair, count the number of relevant documents, a, in that sample, and divide by the number of documents, n, in the sample:

$$\hat{p} = \frac{a}{n} \qquad (7)$$

This is called *simple random sampling.* A more complex approach to random sampling often has advantages, as described in the next section.

5.2.1 Stratified Random Sampling

Simple random sampling is not the only way to estimate a proportion. In *stratified sampling* we use additional knowledge about a population to divide the population into groups or *strata* [1, p. 89]. We then take a simple random sample separately from each stratum, estimate the quantity of interest for each stratum, and combine the stratum estimates to get an overall estimate for the population. If the strata are chosen so that items in a stratum are similar to each other the accuracy of an stratified estimate can be greater than the accuracy of an estimate based on simple random sampling from the whole population.

We stratified the set of filtering documents submitted by each TREC-4 site according to which of the three runs each document was submitted for. By considering all combinations of presence and absence of a document in the three submitted sets, we get 8 strata, as shown in Figure 1.

			Stratum	Number of Documents
R_1	R_2	R_3	Name (h)	in Stratum (N_h)
0	0	0	000	very many
0	0	1	001	many
0	1	0	010	very few or none
0	1	1	011	some
1	0	0	100	very few or none
1	0	1	101	very few or none
1	1	0	110	very few or none
1	1	1	111	few

Figure 1: The test documents submitted by a site can be separated into eight strata, based on which of the three submitted sets, the R_1 set, the R_2 set, and the R_3 set each document appeared in. We indicate presence in the set by a 1, absence by a 0. The comments indicate the relative sizes of the sets in typical filtering track submissions.

Strata 010, 100, 101, and 110 will usually be empty, since in most cases the submitted sets will be such that the R_1 set is contained in the R_2 set, and the R_2 set is contained in R_3 set. In general, however, each of the submitted sets is the union of four strata:

$$\begin{aligned} \text{Set for run } R_1 &\ :\ 100, 101, 110, 111 \\ \text{Set for run } R_2 &\ :\ 010, 011, 110, 111 \\ \text{Set for run } R_3 &\ :\ 001, 011, 101, 111 \end{aligned}$$

To estimate the proportion p_i of relevant documents in set R_i by stratified sampling, we separately estimate the proportion p_h for each stratum h in the R_i set. We then add up the estimated stratum proportions, weighting them by the relative size of their stratum in the submitted set [1, p. 91]:

$$\hat{p}_i \;=\; \textstyle\sum_{h \in R_i} \frac{N_h}{N_i} \hat{p}_h \tag{8}$$

Here h ranges over the strata that make up the R_i set, N_h is the size of stratum h, N_i is the size of the R_i set, and $\hat{p}_h$ is an estimate of the proportion of relevant in stratum h. Expanding this out for runs R_1 to R_3 gives:

$$\hat{p}_1 \;=\; \frac{N_{100}}{N_1} \times \hat{p}_{100} + \frac{N_{101}}{N_1} \times \hat{p}_{101} + \frac{N_{110}}{N_1} \times \hat{p}_{110} + \frac{N_{111}}{N_1} \times \hat{p}_{111} \tag{9}$$

$$\hat{p}_2 \;=\; \frac{N_{010}}{N_2} \times \hat{p}_{010} + \frac{N_{011}}{N_2} \times \hat{p}_{011} + \frac{N_{110}}{N_2} \times \hat{p}_{110} + \frac{N_{111}}{N_2} \times \hat{p}_{111} \tag{10}$$

$$\hat{p}_3 \;=\; \frac{N_{001}}{N_3} \times \hat{p}_{001} + \frac{N_{011}}{N_3} \times \hat{p}_{011} + \frac{N_{101}}{N_3} \times \hat{p}_{101} + \frac{N_{111}}{N_3} \times \hat{p}_{111} \tag{11}$$

These estimates will be unbiased (see Section 7) if the estimate $\hat{p}_h$ of the proportion of relevant documents in stratum h is unbiased for each component stratum h. As our estimate $\hat{p}_h$ we used

	Submitted Docs			Samples from Strata		
Stratum	Rels	NonRels	True Prop	Rels	NonRels	Est. Prop
000	50	100000	.0005	0	0	–
001	20	980	.0200	1	29	.0333
010	2	8	.2000	2	8	.2000
011	80	120	.4000	10	20	.3333
100	0	0	–	0	0	–
101	0	0	–	0	0	–
110	0	0	–	0	0	–
111	30	10	.7500	23	7	.7667
R1 Total	30	10	.7500	–	–	.7667
R2 Total	112	138	.4480	–	–	.3973
R3 Total	130	1110	.1048	–	–	.1054
Test Set Total	182	101118	.0018	–	–	–

Figure 2: Hypothetical data on a site's submitted sets for a single topic. We show both the true and sampled values the number of relevant and nonrelevant documents in each stratum and run, and the corresponding proportion of relevant.

the proportion of relevant documents found in a simple random sample from stratum h:

$$\hat{p}_h = \frac{a_h}{n_h}$$

where n_h is the size of the simple random sample taken from stratum h and a_h is the number of relevant documents found in that sample. This $\hat{p}_h$ is an unbiased estimate of p_h [1, p. 51].

5.3 Sample Sizes in Stratified Sampling for TREC-4

To be consistent with the TREC-4 routing evaluation, at most 100 documents were judged from the three sets of documents submitted by a site for each filtering topic. These 100 documents had to be allocated to as many as seven strata (all except stratum 000), as described above. This was done by choosing equal sized samples from all nonempty strata. If all documents from a stratum were used up by this procedure the leftover documents were allocated equally among the other strata, until a total of 100 was reached, or all documents from the three submitted sets were selected.

6 Stratified Sampling: An Example

Figure 2 displays data on the submitted sets from a hypothetical TREC-4 filtering site for a single topic. The R_2 set is bigger than the R_1 set, and the R_3 set is bigger than both the R_1 and R_2 sets. One anomaly is that 10 documents are in R_2 set but not in R_3 set. This might happen due to a mistake by the site, or because documents were retrieved by boolean queries which were not in a strict generalization relationship.

6.1 Estimating Proportion of Relevant Documents

If all submitted documents were judged, then we could compute the true proportion of relevant documents in each submitted set:

$$p_1 = \frac{30}{30+10} = .7500 \tag{12}$$

$$p_2 = \frac{112}{112+138} = .4480 \tag{13}$$

$$p_3 = \frac{130}{130+1110} = .1048 \tag{14}$$

To compute a stratified estimate of these proportions, we assume that simple random samples were drawn from each stratum and judged for relevance, as shown in Figure 2. This gives estimates of the proportion of relevant documents in each stratum, as shown in the last column of Figure 2. We then combine the stratum estimates, using Equations 9 to 11, to get estimates of the proportion of relevant in each submitted set:

$$\hat{p}_1 = \frac{40}{40} \times \frac{23}{23+7} = .7667 \tag{15}$$

$$\hat{p}_2 = \frac{10}{250} \times \frac{2}{2+8} + \frac{200}{250} \times \frac{10}{10+20} + \frac{40}{250} \times \frac{23}{23+7} = .3973 \tag{16}$$

$$\hat{p}_3 = \frac{1000}{1240} \times \frac{1}{1+29} + \frac{200}{1240} \times \frac{10}{10+20} + \frac{40}{1240} \times \frac{23}{23+7} = .1054 \tag{17}$$

6.2 Estimating Utility of a Submitted Set

If we knew the true proportion of relevant documents in each submitted set (Equations 12 to 14), we could compute the true utility of each set, using Equations 2 to 4:

$$u_1 = (4 \times .7500 - 3) \times 40 = 0.0 \tag{18}$$

$$u_2 = (2 \times .4480 - 1) \times 250 = -26.0 \tag{19}$$

$$u_3 = (4 \times .1048 - 1) \times 1240 = -719.2 \tag{20}$$

If we instead have the stratified estimates of the proportions, we use them to get estimates of the total utility:

$$\hat{u}_1 = (4 \times .7667 - 3) \times 40 = 2.672 \tag{21}$$

$$\hat{u}_2 = (2 \times .3973 - 1) \times 250 = -51.35 \tag{22}$$

$$\hat{u}_3 = (4 \times .1054 - 1) \times 1240 = -717.2 \tag{23}$$

The estimates for the R_1 and R_3 sets are close to the true values, while the estimate for the R_2 set is less close. We can see why by comparing in Figure 2 the true and estimated proportion of relevant for each stratum in the R_2 set. Due to bad luck with our random sample from 011, the largest stratum in the R_2 set, we underestimated the proportion of relevant in that stratum. This carried over to our estimate of the overall proportion of relevant for the R_2 set, and thus to the total utility. This raises the question of how much confidence we can have in our estimates of utility, and is the subject of the next section.

7 How Accurate are Our Estimates of Utility?

The pooling and stratified sampling approaches are based on judging only a subset of each submitted set, so in neither case will the estimates of utility be perfect. A common measure of the distance between an estimate, $\hat{\mu}$, and the quantity we want to estimate, μ, is the mean square error (MSE) [1, p. 15]. The MSE of an estimate is the expected value of the square of the difference between the estimate and the true value:

$$\mathrm{MSE}[\hat{\mu}] = \mathrm{E}[(\hat{\mu} - \mu)^2] \tag{24}$$

Letting $m = \mathrm{E}[\hat{\mu}]$, the MSE can be rewritten as the sum of two terms:

$$\begin{aligned}
\mathrm{MSE}[\hat{\mu}] &= \mathrm{E}[(\hat{\mu} - \mu)^2] \\
&= \mathrm{E}[((\hat{\mu} - m) - (\mu - m))^2] \\
&= \mathrm{E}[(\hat{\mu} - m)^2 - 2(\mu - m)(\hat{\mu} - m) + (\mu - m)^2] \\
&= \mathrm{E}[(\hat{\mu} - m)^2] - \mathrm{E}[2(\mu - m)(\hat{\mu} - m)] + \mathrm{E}[(\mu - m)^2] \\
&= \mathrm{E}[(\hat{\mu} - m)^2] - 2 \times 0 \times \mathrm{E}[(\hat{\mu} - m)] + (\mu - m)^2 \\
&= \mathrm{E}[(\hat{\mu} - m)^2] + (\mu - m)^2 \\
&= \mathrm{E}[(\hat{\mu} - \mathrm{E}[\hat{\mu}])^2] + (\mu - \mathrm{E}[\hat{\mu}])^2 \\
&= \mathrm{Var}[\hat{\mu}] + \mathrm{Bias}[\hat{\mu}].
\end{aligned}$$

The first term:

$$\mathrm{Var}[\hat{\mu}] = \mathrm{E}[(\hat{\mu} - \mathrm{E}[\hat{\mu}])^2]$$

is the *variance* of the estimator $\hat{\mu}$ and measures the tendency of the estimator to deviate from its own expected value. The second term:

$$\mathrm{Bias}[\hat{\mu}] = (\mathrm{E}[\hat{\mu}] - \mu)^2$$

is the *bias* of $\hat{\mu}$ and measures the systematic difference between the expected value of the estimator and the value we are trying to estimate. It is often, though not always, desirable to use *unbiased* estimates of a quantity. An estimate is unbiased if $\mathrm{E}[\hat{\mu}] = \mu$, i.e. $\mathrm{Bias}[\hat{\mu}] = 0$.

These two concepts, bias and variance, and their sum the MSE, will be useful in discussing the accuracy of our estimates of utility.

7.1 Accuracy of Pooled Estimates

The MSE of the pooled estimates is difficult to determine, since the pool is not constructed randomly. The variance of a pooled estimate is nonzero, since we do not sample the entire population. However, the variance is likely to be smaller than that of the corresponding stratified estimate, due to the large number of documents judged.

The pooled estimate has a nonzero bias as well, since if there are any relevant documents in the submitted set which were not judged, the estimated utility will be lower than the true utility. In fact, not only is the expected value of the pooled estimate always less than the true utility, but the actual value of the pooled estimate is always less than the true utility. So the pooled estimate is a lower bound on the true utility.

7.2 Accuracy of Estimates Based on Random Sampling

Recall that the utility of a submitted set can be expressed in terms of the proportion of relevant documents in that set:

$$u_i = ((u_{ai} - u_{bi})p_i + u_{bi})N_i \tag{25}$$

Similarly, we can estimate the utility of a submitted set based on an estimate of the proportion of relevant documents in that set:

$$\hat{u}_i = ((u_{ai} - u_{bi})\hat{p}_i + u_{bi})N_i \tag{26}$$

The MSE of such an estimate is

$$\begin{aligned}
\mathrm{MSE}[\hat{u}_i] &= \mathrm{E}[(\hat{u}_i - u_i)^2] \\
&= \mathrm{E}[(((u_{ai} - u_{bi})\hat{p}_i + u_{bi})N_i - ((u_{ai} - u_{bi})p_i + u_{bi})N_i)^2] \\
&= \mathrm{E}[(u_{ai} - u_{bi})^2 {N_i}^2 (\hat{p}_i - p_i)^2] \\
&= (u_{ai} - u_{bi})^2 {N_i}^2 \mathrm{E}[(\hat{p}_i - p_i)^2] \\
&= (u_{ai} - u_{bi})^2 {N_i}^2 \mathrm{MSE}[\hat{p}_i]. \qquad (27)
\end{aligned}$$

So the MSE of $\hat{u}_i$ is a simple function of the MSE of our estimate of the proportion of relevant documents. For simple random sampling and stratified sampling, the estimates of the proportion are unbiased, that is $\mathrm{E}[\hat{p}_i] = p_i$. Therefore, the MSE of $\hat{p}_i$ results solely from its variance, and we have:

$$\mathrm{MSE}[\hat{u}_i] = (u_{ai} - u_{bi})^2 {N_i}^2 \mathrm{Var}[\hat{p}_i]. \tag{28}$$

Also note that $\hat{u}_i$ is unbiased as well, so its MSE consists solely of variance.

In the rest of this section we will look at what $\hat{p}_i$'s variance is under different sampling techniques.

7.2.1 Variance of Proportions Estimated by Simple Random Sampling

We begin with the estimate produced by simple random sampling, as this is both a component of, and a point of comparison with, the stratified sampling method used for the filtering evaluation. Recall that our estimator of the proportion of relevant documents, based on a simple random sample from a set, is:

$$\hat{p} = \frac{a}{n} \tag{29}$$

where n is the size of the simple random sample, and a is the number of relevant documents in the sample. We cannot know the exact variance of this estimate without knowing the actual value of p, which is of course what we are trying to estimate in the first place. However, an unbiased estimate of the variance of our estimate of the proportion is [1, p. 52]:

$$\begin{aligned}\widehat{\mathrm{Var}}[\hat{p}] &= \frac{N-n}{(n-1)N} \times \hat{p} \times (1-\hat{p}) \\ &= \frac{N-n}{(n-1)N} \times \frac{a}{n} \times \frac{n-a}{n} \\ &= \frac{(N-n)a(n-a)}{n^2(n-1)N} \end{aligned} \tag{30}$$

Suppose we used a simple random sample from set R_i to estimate the utility of set R_i. Then the MSE of the resulting utility estimate for set R_i would have been:

$$\mathrm{MSE}[\hat{u}_i] = (u_{ai} - u_{bi})^2 \frac{(N_i - n_i)a_i(n_i - a_i)}{{n_i}^2(n_i - 1)N_i} \tag{31}$$

7.2.2 Variance of Proportions Estimated by Stratified Sampling

In stratified sampling we separately estimate the proportion of relevant in each stratum and combine these estimates to get an estimate of the overall proportion:

$$\hat{p}_i = \sum_{h \in R_i} \frac{N_h}{N_i} \hat{p}_h \tag{32}$$

By the properties of the variance of linear combinations of random variables, and the fact that our samples from the strata are independent, we have [1, p. 92]:

$$\mathrm{Var}[\hat{p}_i] = \sum_{h \in R_i} \frac{{N_h}^2}{{N_i}^2} \mathrm{Var}[\hat{p}_h] \tag{33}$$

Each $\hat{p}_h$ is an estimate of the proportion of relevant in a stratum, based on a simple random sample from the stratum. Therefore, the results of the previous section tell us that an unbiased estimate of the variance of $\hat{p}_h$ is:

$$\widehat{\mathrm{Var}}[\hat{p}_h] = \frac{(N_h - n_h)a_h(n_h - a_h)}{{n_h}^2(n_h - 1)N_h} \tag{34}$$

Substituting Equation 34 into Equation 33 then gives us an unbiased estimate of the variance of our stratified estimate of the proportion of relevant in the R_i set:

$$\begin{aligned}\widehat{\mathrm{Var}}[\hat{p}_i] &= \sum_{h \in R_i} \frac{{N_h}^2}{{N_i}^2} \frac{(N_h - n_h)a_h(n_h - a_h)}{{n_h}^2(n_h - 1)N_h} \\ &= \frac{1}{{N_i}^2} \sum_{h \in R_i} \frac{N_h(N_h - n_h)a_h(n_h - a_h)}{{n_h}^2(n_h - 1)} \end{aligned} \tag{35}$$

Further substituting Equation 35 into Equation 28 gives us the MSE for the estimate $\hat{u}_i$ (Equation 26) based on the stratified estimate of $\hat{p}_i$:

$$\begin{aligned} \mathrm{MSE}[\hat{u}_i] &= (u_{ai} - u_{bi})^2 {N_i}^2 \mathrm{Var}[\hat{p}_i] \\ &= (u_{ai} - u_{bi})^2 {N_i}^2 \frac{1}{{N_i}^2} \sum_{h \in R_i} \frac{N_h(N_h - n_h)a_h(n_h - a_h)}{{n_h}^2(n_h - 1)} \\ &= (u_{ai} - u_{bi})^2 \sum_{h \in R_i} \frac{N_h(N_h - n_h)a_h(n_h - a_h)}{{n_h}^2(n_h - 1)} \end{aligned} \tag{36}$$

If we compare Equation 36 to Equation 31, we see that the stratified estimate has a smaller MSE than an estimate based on simple random sampling when:

$$\sum_{h \in R_i} \frac{N_h(N_h - n_h)a_h(n_h - a_h)}{{n_h}^2(n_h - 1)} < \frac{(N-n)a(n-a)}{n^2(n-1)N}$$

This is almost always true when reasonable strata are defined and appropriately sized samples are chosen from those strata [1, p. 99].

8 Stratified Sampling: An Example (Part II)

Returning to our example, we can use Equation 36 to give the MSE's of the utility estimates in Equations 21–23:

$$\mathrm{MSE}[\hat{u}_1] = 4^2(0 + 0 + 0 + \frac{64400}{26100}) = 39.5 \tag{37}$$

$$\mathrm{MSE}[\hat{u}_2] = 2^2(0 + \frac{6800000}{26100} + 0 + \frac{64400}{26100}) = 1052.0 \tag{38}$$

$$\mathrm{MSE}[\hat{u}_3] = 4^2(\frac{28130000}{26100} + \frac{6800000}{26100} + 0 + \frac{64400}{26100}) = 21452.5 \tag{39}$$

Recall that the $\hat{u}_i$ are unbiased, so the MSE of each estimate is just its variance, i.e. $\mathrm{MSE}[\hat{u}_1] = \mathrm{Var}[\hat{u}_i]$. Making the reasonable assumption that $\hat{u}_i$ has a roughly normal distribution, then a 95% confidence interval around $\hat{u}_i$ is [8, ch. 7]:

$$\hat{u}_i \pm 1.96\sqrt{\mathrm{Var}[\hat{u}_i]}.$$

Then combining Equations 21–23 with Equations 37–39, and using the above expression for the confidence interval gives:

$$\begin{aligned} u_1 &= 2.672 \pm 12.3 \\ u_2 &= -51.35 \pm 63.6 \\ u_3 &= -717.2 \pm 287.1 \end{aligned}$$

We of course arranged this example so that the true utilities (Equations 18 to 20), which are known in our example but which would not be known in general, fell within the 95% confidence intervals. This would usually be the case in practice.

9 Discussion

The results of the TREC-4 filtering evaluation appear in an appendix to these proceedings. Table 1 for each site shows the raw data used in computing utility estimates. For each topic and each of the three runs, we see the number of documents submitted and the pooled and stratified estimates of the utility of those submitted sets. Additional tables provide both summary and graphical presentations of this data.

One reassuring observation is that the pooled estimates in Table 1 are almost always within the confidence intervals for the stratified estimates. This suggests that the bias of the pooled estimates is not large in a statistical sense. Since the pooled utility provides a lower bound on the true utility, combining the two estimates can be useful. For instance, Topic 45, Run 2 for the *pir* (Queen's) system has a stratified estimate of 16.8 ± 12.8, but the pooled estimate of 16.0 shows that the true utility must be in the upper part of this range.

In many cases the confidence intervals have a margin of error [8, p. 451] of ± 0.0. This can occur for two very different reasons. First, if the submitted set was small enough that the entire set could be included in the stratified sample, we in fact have an exact value for that set's utility, not an estimate, and so there is no error possible.

In contrast, a margin of error of ± 0.0 can also arise because the variances which enter into the confidence intervals are themselves estimates. In particular, if the random sample from a stratum contains no relevant documents or only relevant documents, the estimated variance will be 0.0.

For example, Topic 30, Run 3 for Xerox had a submitted set of size 561 and an estimated utility of -235.3 ± 144.5. The margin of error is large because relatively few documents from the set could be judged. Topic 50, Run 3 for Xerox also had a large submitted set. Its margin of error is given as ± 0.0 because no relevant documents appeared in the stratified sample. While 0.0 is an unbiased estimate of the variance, we can rightly be skeptical of this value. A more extreme case is Topic 142, Run 3 for HNC where the confidence interval for the stratified estimate is -47.0 ± 0.0. However, the pooled estimate, which is a lower bound on the true utility, is 33.0, well outside the (degenerate) confidence interval.

In most cases, however, the margins of error are both small and believable. The largest submitted set was 809 documents, putting to rest worries that submitted sets with sizes in the thousands would cause both statistical and practical problems for the evaluation.

All sites attempted to adjust the size of submitted sets to reflect the utility measures for the three runs. As the papers from the sites discuss, doing this in an optimal fashion is difficult and is likely to be the focus of future research. Achieving good results in run R_1, where a precision of over 0.75 was required to achieve a positive utility, was particularly difficult. All sites had 30 or more topics where they would have been better off submitting no documents than submitting their run R_1 set.

Our policy of drawing the stratified sample evenly from all strata (see Section 5.3) resulted in relatively tight confidence intervals on the run R_1 and R_2 stratified estimates, and relatively loose intervals on run R_3. In future evaluations, it may be desirable to draw more documents from the run R_3 strata, to make the confidence intervals somewhat more balanced.

10 Summary

The TREC-4 filtering track broke new ground for TREC by introducing a binary classification task and a set of new evaluation procedures. The sites who participated in the track (HNC, NSA, Queen's, and Xerox), had to tolerate considerable uncertainties and are to be congratulated for their perseverance. The filtering track will be run again in TREC-5, with much the same structure as in TREC-4, and we encourage sites to take part.

11 Acknowledgments

I am greatly appreciative of Donna Harman and her team at NIST, for both their work in conducting the filtering evaluation and their suggestions for improving it. Gavin O'Brien in particular put great effort into generating and presenting the filtering results. The ideas presented here were developed in extensive discussions with the members of the TREC-1 through TREC-4 program committees, and with TREC-4 participants, particularly David Hull, Paul Kantor, K. L. Kwok, and Julian Yochum.

References

[1] William G. Cochran. *Sampling Techniques.* John Wiley & Sons, New York, 3rd edition, 1977.

[2] William S. Cooper. On selecting a measure of retrieval effectiveness. *Journal of the American Society for Information Science*, 24:87–100, March–April 1973.

[3] Donna Harman. Overview of the fourth Text REtrieval Conference (TREC-4). In D. K. Harman, editor, *The Fourth Text REtrieval Conference (TREC-4)*, Gaithersburg, MD, 1996. U. S. Dept. of Commerce, National Institute of Standards and Technology.

[4] Paul S. Jacobs. GE in TREC-2: Results of a boolean approximation method for routing and retrieval. In D. K. Harman, editor, *The Second Text Retrieval Conference (TREC-2)*, pages 191–199, Gaithersburg, MD, March 1994. U. S. Dept. of Commerce, National Institute of Standards and Technology. NIST Special Publication 500-215.

[5] David D. Lewis. Evaluating text categorization. In *Proceedings of Speech and Natural Language Workshop*, pages 312–318. Defense Advanced Research Projects Agency, Morgan Kaufmann, February 1991.

[6] David D. Lewis. Evaluating and optimizing autonomous text classification systems. In Edward A. Fox, Peter Ingwersen, and Raya Fidel, editors, *SIGIR '95: Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 246–254, New York, 1995. Association for Computing Machinery.

[7] David D. Lewis and William A. Gale. A sequential algorithm for training text classifiers. In W. Bruce Croft and C. J. van Rijsbergen, editors, *SIGIR 94: Proceedings of the Seventeenth*

Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, pages 3–12, London, 1994. Springer-Verlag.

[8] David S. Moore and George P. McCabe. *Introduction to the Practice of Statistics.* W. H. Freeman, New York, 1989.

[9] S. E. Robertson. The probability ranking principle in IR. *Journal of Documentation*, 33(4):294–304, December 1977.

[10] Richard M. Tong and Lee A. Appelbaum. Machine learning for knowledge-based document routing (a report on the TREC-2 experiment). In D. K. Harman, editor, *The Second Text Retrieval Conference (TREC-2)*, pages 253–264, Gaithersburg, MD, March 1994. U. S. Dept. of Commerce, National Institute of Standards and Technology. NIST Special Publication 500-215.

Using Relevance Feedback and Ranking in Interactive Searching

Nicholas J. Belkin[†], Colleen Cool, Jürgen Koenemann[*], Kwong Bor Ng, Soyeon Park

School of Communication, Information & Library Studies
Rutgers University
4 Huntington Street
New Brunswick, NJ 08901-1071
[nick@belkin,cool@zodiac,koeneman@ruccs,kng@zodiac,ypark@zodiac].rutgers.edu

Abstract

We present results of a study in which 50 searchers, of varying degrees of experience in information retrieval (IR), each performed searches on two TREC-4 adhoc interactive track topics, using a simple interface to the INQUERY retrieval engine. The foci of our study were: the relationships between the users' models and experience of IR, and their performance in the TREC-4 adhoc task while using a best-match IR system with relevance feedback; the understanding, use and utility of relevance feedback and ranking in interactive IR; and, the evaluation of interactive IR.[†] [*]

1. Introduction

In this paper, we report on some preliminary results of our TREC-4 study. We first present a description of our investigative design, then present and discuss some results from the study, and finally draw some conclusions about the presentation and use of relevance feedback and ranking for end users, and about the evaluation of interactive IR.

In our TREC-3 study (Koenemann et al., 1995), we noted that there appeared to be a relationship between the kind and strength of the model of IR held by our expert searchers, and their behavior (both performance in terms of effectiveness measures, and patterns of interaction in their searches) when constructing a routing query using a system which offered ranked output and relevance feedback, in a full-text environment. But because we had relatively little information about their models of IR, and because the searchers were relatively homogeneous in terms of their experience with IR systems, we were unable to follow up on this observation. We therefore designed our TREC-4 study explicitly to consider this issue, and also to consider the related issue of how people with little or no experience of ranked output, relevance feedback IR would understand and use these features.

2. Methods

2.1 Subjects

We recruited fifty volunteers to participate as searchers in our study. The subjects were chosen by circulating a recruitment announcement among the faculty and graduate students of the School of Communication, Information & Library Studies at Rutgers University, among the community of information professionals in New Jersey, and to several information-intensive research companies in New Jersey. The general demographic characteristics of the volunteers are given in section II of the Interactive System Description, together with various measures of their experience with IR systems. Three of the volunteers had also participated in our TREC-3 study, and therefore had some experience with the INQUERY retrieval engine, although not with the specific interface and features which we used in TREC-4.

2.2 System

We used the INQUERY retrieval engine (version 2.1p3) (cf. Callan, et al., 1992) with an interface based on, but somewhat modified from, the standard X-Windows interface released with that version of INQUERY. Figure 1 (in section I of the Interactive System Description) is a screen dump of the interface that we used. Because we were interested explicitly in how previous experience in IR related to the use of the novel functions of relevance feedback and ranking, we restricted query formulation to unstructured input, without any of the operators offered by INQUERY, with the exception of the ability to specify adjacent-word phrases. The general functions offered by this interface are: unstructured query input; the ability to save a query and to recall a saved query (either on its own, or in combination with other queries); the ability to mark (and unmark) a document as relevant, for relevance feedback purposes; the ability to mark (and unmark) a document as saved (separate from marking it relevant); a scrollable display of ten document titles with document IDs which indicate the source of the document and with the rank-order of each document, but not its retrieval status value (only the top 150 documents were actually returned to the user); the

[†] To whom all correspondence should be addressed.

[*] Rutgers University Center for Cognitive Science

ability to call and scroll through the full text of any specified document from the list of retrieved documents; query terms (including terms added by relevance feedback) indicated by highlighting in the display of the text of the retrieved documents; the number of terms (but not the terms themselves) added to a query as the result of relevance feedback indicated in a line below the query formulation window. See Section I of the Interactive System Description for how these features are implemented in the system. Relevance feedback in INQUERY version 2.1p3 allows only positive relevance judgments, and adds five terms to the query if only one document is marked relevant, and two more for each subsequent relevant document. The reweighting and term-selection is done according to the optimum formulae as described in Haines & Croft (1993). The system ran on a dedicated SPARCStation 5 with a seventeen-inch color monitor (see the System Description for the Rutgers Interactive Track Study for details).

2.3 Procedures

An appointment was made with each volunteer searcher to come to the experiment site for a two and one-quarter hour session. Each session consisted of: a pre-search questionnaire and a pre-search interview (together lasting about 15 minutes); a hands-on tutorial for our version of INQUERY (about 40 minutes); a practice search using TREC-4 adhoc topic 224, to familiarize the subjects with INQUERY and with the task which they would be set (15 minutes); two searches on two of the 25 TREC-4 interactive adhoc topics (30 minutes each); a brief search-evaluation questionnaire administered after each search; and, a closing interview (about 15 minutes). The two interviews were tape-recorded, the tutorial, practice search, and searches were completely logged (all actions passed between the interface and the search engine were recorded and time-stamped), the subjects were asked to think-aloud during the three searches, and both their thinking-aloud and their interactions with the interface (i.e. the monitor screen) were recorded on videotape. One or two investigators were present with each subject throughout the entire process, to conduct the interviews, to introduce the tutorial and the search tasks, and to be available in case of system failure or other problem. Explanation of system features was left to the tutorial only. See Appendix A for the questions that were asked in the various questionnaires and interviews, and the Interactive System Description for a summary of the tutorial.

The task that the subjects were set was a slight variation of the standard TREC-4 interactive task specification. Subjects, seated in front of the computer with which they would be interacting, were handed a form (Interactive System Description, section II.2), which asked them to imagine that they were themselves interested in the information problem described there (one of the interactive adhoc topics). On this form, they were told that their task was to find as many good documents as they could on this topic, in up to 30 minutes (15 minutes for the practice search). They were told that they could indicate good documents by saving them explicitly, by constructing a query which they thought would get a lot of good documents at the top of the list, or both methods in combination. The investigator went through this form with the subject, and the search was deemed to have started at the time that the subject received the form. Subjects were warned after 25 minutes of searching, and asked to start to finish up if they were still searching at 30 minutes.

Each subject searched on two of the 25 interactive adhoc topics, which means that there were four separate searches, each by a different searcher, for each topic. To control for possible learning effects, the order of presentation of the topics was varied so that each topic was twice the first one given to a searcher, and twice the second. Topic presentation was also varied so that no two searchers were given the same two topics to search, in either order. Because the interactive adhoc topics were chosen randomly, with respect to generators of the topics, from the full set of adhoc topics, and because we had no prior information about characteristics of these topics such as difficulty, we made no attempt to control the distribution of topics on any such features.

3. Results

3.1 Effectiveness

We submitted three sets of retrieval results for our searchers. The "official primary" result consisted of the documents "saved"[1] by one searcher for each topic, ranked in the order in which they were saved. The "official secondary" result consisted of the top 1000 documents retrieved by the "final query" specified by the same searcher for each topic (three topics had no final queries saved), with the explicitly saved documents at the top of the list (frozen rank). These two results are indicated, respectively as ruint1/q0 and ruint1/q1. The searches which were included in ruint1 were chosen according to the following general rules:

1. choose a search which has a "final" query, or more than 30 "saved" queries;
2. prefer a second search to a first search (on the grounds of learning effect);

[1] Because of the way that we specified the task to our users, we define "saved" as meaning all of the documents which a searcher has explicitly marked as "saved", plus, if this were less than 30 documents, enough documents from the top of the list of those retrieved by any query that the searcher saved as a "final" query to make 30 "saved" documents total. In such combined cases, explicitly saved documents are ranked ahead of those implicitly saved by the final query.

3. if more than one search meets these conditions, choose that with more "saved" queries.

In order to maximize the number of documents judged, we submitted a third set of results (ruint2) which combined, for each topic, all of the documents which were explicitly saved by all of the other three searchers for that topic, plus the top 1000 documents retrieved by any saved "final" query (these lists appropriately merged). Because of the way that these lists of documents were generated, we do not discuss any evaluation based on them.

As a rough means of comparison between interactive and non-interactive searching, we also submitted one set of "baseline" results (ruibsl). These were produced by submitting the entire text of the topic description (edited to remove symbols such as "/" and "()" which cause problems for the INQUERY query parser) as an unstructured query to the standard INQUERY retrieval engine.

	ruibsl	ruint1/q1
Recall/Prec		
0.00	0.5884	0.9107
0.10	0.3555	0.6251
0.20	0.3082	0.4560
0.30	0.2587	0.3269
0.40	0.2292	0.2689
0.50	0.1738	0.2288
0.60	0.1251	0.1776
0.70	0.0800	0.1227
0.80	0.0645	0.0659
0.90	0.0100	0.0282
1.00	0.0000	0.0081
Avg Prec	0.1808	0.2668
Prec@30	0.2733	0.3269
R-Prec	0.2287	0.2668

Table 1. Comparison of performance of baseline non-interactive searching (ruibsl) with the secondary task of saving a final query (ruint1/q1) for one set of interactive searches per topic, top 1000 documents.

The evaluations of the retrieval results for ruint1/q1 and ruibsl, according to the standard TREC evaluation measures, are presented in Table 1. It should be noted here that although average precision over the top 1000 documents is significantly higher for the interactive searches, the difference between precision at 30 documents is rather less, as is that of R-Precision. Table 1 compares the results of only one interactive search for each topic with the one baseline search. In addition, three of the interactive searches in this set had no final query, which leads to some problem with interpretation of the comparison. In Table 2, we compare performance between the interactive and non-interactive systems when only the top 30 or fewer documents retrieved are considered, for *all* interactive searches, not just those submitted as official results (the restriction to the top 30 is because many interactive searches led to 30 or fewer saved documents). The comparison is between the performance for each searcher and the baseline, for 100 datapoints, which allows us to consider the numbers of interactive searches which performed below, at, or better than the baseline non-interactive search.

	Mean Diff	Below Bsl	At Bsl	Above Bsl
Prec.	+0.247*	13	19	68
Recall	-0.013	24	51	25

*Significant at $p \leq .001$

Table 2. Comparison of all interactive searches for secondary task against baseline search (Bsl), all at cutoff of 30 documents. Below, at and above Bsl is the number of interactive searches at that level.

Topic	Ret	Rel	Rel Ret	Prec	Recall	Time
202	44	283	32	0.7273	0.1131	33
203	30	33	5	0.1667	0.1515	22
204	30	397	26	0.8667	0.0655	23
205	30	310	16	0.5333	0.0516	28
206	30	47	10	0.3333	0.2128	28
207	30	74	20	0.6667	0.2703	32
208	30	54	7	0.2333	0.1296	30
209	14	87	11	0.7857	0.1264	32
210	32	57	26	0.8125	0.4561	32
211	17	323	17	1.	0.0526	32
212	30	153	24	0.8	0.1569	30
213	30	21	6	0.2	0.2857	19
214	30	5	4	0.1333	0.8	28
215	30	183	23	0.7667	0.1257	20
216	32	36	19	0.5938	0.5278	34
220	30	24	13	0.4333	0.5417	17
223	20	363	15	0.75	0.0413	32
227	45	347	42	0.9333	0.1210	31
232	30	9	3	0.1	0.3333	34
236	30	43	3	0.1	0.0698	31
238	30	270	23	0.7667	0.0854	33
239	30	123	8	0.2667	0.0650	15
242	30	38	21	0.7	0.5526	30
243	30	69	9	0.3	0.1304	29
250	30	86	15	0.5	0.1744	33
Mean				0.5387	0.2256	28.3
Total	744	3435	398			

Table 3. Performance figures for ruint1/q0, the primary interactive searching task, one searcher per topic.

Because of the nature of the "primary" interactive task, which asks searchers to save some (presumably small) set of good documents, it is inappropriate to evaluate ruint1/q0 according to the TREC 4 standard measures. Table 3 therefore presents precision and recall for the set of documents "saved" by the searcher, which is the "primary" evaluation of this task (together with the time taken to perform the search). Because of the great differences in numbers of relevant documents for the different topics, in Table 3 and all subsequent performance evaluation tables for the primary task, we use *macro-averaged* (mean of the individual ratios for each topic) recall and precision.

Precision and Recall for Explicitly or Implicitly Saved Documents											
Topic	Num Rel	Lowest Precision Search 1		Search 2		Search 3		Highest Precision Search 4		Average	
		Prec.	Recall	Prec.	Recall	Prec.	Recall	Prec.	Recall	Prec.	Recall
202	283	0.0067	0.0071	0.7273	0.1137	0.8000	0.0707	0.8667	0.0919	0.6152	0.0709
203	33	0.0000	0.0000	0.0769	0.0303	0.1667	0.1515	0.5000	0.0606	0.1859	0.0606
204	397	0.0000	0.0000	0.2000	0.0025	0.7500	0.0076	0.8667	0.0655	0.4542	**0.0189**
205	310	0.1333	0.0129	0.5333	0.0516	0.5667	0.0548	0.9000	0.0290	0.5333	0.0371
206	47	0.1667	0.1064	0.3333	0.2128	0.5000	0.0851	1.0000	0.0638	0.5000	0.1103
207	74	0.6667	0.1081	0.6667	0.2703	0.7000	0.2838	0.7333	0.2973	0.6917	0.2399
208	54	0.0333	0.0185	0.2333	0.1296	0.2353	0.0741	0.5000	0.0556	0.2505	0.0695
209	87	0.0000	0.0000	0.3125	0.0575	0.4667	0.1609	0.7857	0.1264	0.3912	0.0862
210	57	0.8125	0.4561	0.8571	0.2105	0.9167	0.3860	0.9474	0.3158	0.8834	0.3421
211	323	0.9167	0.0341	1.000	0.0279	1.0000	0.0433	1.0000	0.0526	**0.9792**	0.0395
212	153	0.4286	0.0196	0.7500	0.0196	0.8000	0.1569	1.0000	0.0588	0.7447	0.0637
213	21	0.2000	0.2857	0.2000	0.2857	0.3333	0.0952	0.3846	0.2381	0.2795	0.2262
214	5	0.0333	0.2000	0.1333	0.8000	0.5000	0.6000	1.0000	0.4000	0.4167	**0.5000**
215	183	0.6667	0.1093	0.7000	0.1148	0.7667	0.1257	0.8333	0.1366	0.7417	0.1216
216	36	0.5938	0.5278	0.6333	0.5278	0.6667	0.5556	0.8000	0.2222	0.6735	0.4584
220	24	0.4333	0.5417	0.7143	0.2083	1.0000	0.1250	1.0000	0.1667	0.7869	0.2604
223	363	0.3333	0.0275	0.6667	0.0551	0.7500	0.0413	0.8462	0.0303	0.6491	0.0386
227	347	0.6333	0.0548	0.8333	0.0720	0.9333	0.1210	0.9565	0.0634	0.8391	0.0778
232	9	0.0000	0.0000	0.1000	0.3333	0.1429	0.1111	0.3333	0.1111	0.1441	0.1389
236	43	0.0000	0.0000	0.0000	0.0000	0.1000	0.0698	0.4000	0.0465	**0.1250**	0.0291
238	270	0.4000	0.0148	0.5333	0.0593	0.6667	0.0741	0.7667	0.0852	0.5917	0.0584
239	123	0.0667	0.0163	0.2333	0.0569	0.2667	0.0650	0.3333	0.0813	0.2250	0.0549
242	38	0.2000	0.1579	0.3667	0.2895	0.5667	0.4474	0.7000	0.5526	0.4584	0.3619
243	69	0.0000	0.0000	0.2333	0.1014	0.2667	0.1159	0.3000	0.1304	0.2000	0.0893
250	86	0.2000	0.0698	0.5000	0.1744	0.6667	0.0233	1.0000	0.0116	0.5917	0.0698
Mean	**137.4**	**0.279**	**0.111**	**0.462**	**0.168**	**0.581**	**0.162**	**0.750**	**0.140**	**0.518**	**0.145**
Median	**74**	**0.200**	**0.027**	**0.500**	**0.114**	**0.667**	**0.111**	**0.833**	**0.085**	**0.533**	**0.085**

Table 4. Recall and precision for each of 25 search topics. Given are precision and recall for the set of explicitly saved documents if more than 30 documents were saved by the searcher, or if less than 30 documents were saved, without a final query. If searchers saved only a final query, and did not explicitly save individual documents, the top 30 documents retrieved by the query were used as the saved set (implicitly saved documents). If searchers saved a final query in addition to explicitly saving less than 30 documents, the top-ranked documents retrieved by the query were added to the saved set, resulting in a retrieved set of 30 documents. The four individual searches performed by four different people are sorted in increasing order by precision. The average worst and best precision and recall results are highlighted. All averages are macro-averages.

Table 4 presents the macro-averaged precision and recall for each of the four separate searches for each separate topic, based on the explicitly saved documents for each search plus enough from any saved final query to reach a minimum of 30. It is thus the equivalent to the data presented in Table 3, but for all of our searchers, rather than for one search per topic, as in ruint1/q0. The data in Table 4 are presented, for each topic, in rough order of performance of the searches, as indicated by precision. Thus, in addition to indicating overall performance, the data in Table 4 are indicative of both differences in performance (perhaps of difficulty) between topics, and of the rather large range of performance by the different searchers within any one topic. The overall precision for all searches is not significantly different from the precision for the set of single searches in ruint1/q0, which suggests that our method of selection for that set led to a reasonable sample from the searches as a whole. Although ruint1/q0 had higher recall than the searchers overall, this is due to having selected searchers for ruint1 who had saved final queries.

The data in Table 3 and especially in Table 4 raise interesting questions about how to evaluate performance for this interactive searching task, and perhaps for interactive searching in general. Although there is a steady and regular increase in overall performance, as measured by mean precision, from the groups of least good to best performing searches, there are anomalies resulting from the differences in number of saved documents. For instance, for topic 220 in Table 4, the "worst" search retrieved 13 relevant documents out of 30 saved documents; the "best" search saved only 4 documents, all of which were relevant. Using van Rijsbergen's E (van Rijsbergen, 1979) for evaluation in this case (with no preference between recall and precision) leads to a substantial benefit for the "worst" search (0.5185) over the "best" (0.7143). Although this particular example may only demonstrate that our classification of searches is faulty, it also suggests that the measures that are being used for the TREC 4 "primary" evaluation of this searching task need to be re-examined.

3.2 Relationships between performance and other variables

We were interested in possible relationships between characteristics of the searchers and their performance, and also in relationships between other characteristics of the searches, and performance. Somewhat surprisingly, *none* of the demographic or experience variables obtained from the pre-search interview is significantly related to performance, as measured in Table 4. Also, there is no significant relationship between time taken for a search and performance, nor between the order of search for a topic and performance. The latter would be an indication of learning effect, but it appears that inter-topic differences are so large that any possible effect is swamped by this factor.

A major concern of our ongoing research is to discover if there are any relationships between the searchers' mental models of IR and their performance in the system, either in terms of search effectiveness measures such as recall and precision, or in terms of other measures, or of search behaviors. This is an area we will explore further with the data we have collected in this study. At this point, we have, on the basis of content analysis of our interview data, been able to identify a set of search strategies, which we believe are reflective of such models, and which appear to have had some relationship to the behaviors in which our searchers engaged. Although these are similar to search strategies and tactics that have been identified by, for instance, Bates (1979) and Hewitt & Scott (1987), there are also some differences. These strategies are indicated in Table 5. Many of our searchers indicated that they were able to use their familiar strategies in this unfamiliar system, by adapting them to the new features. For instance, Restriction was approximated by marking many similar documents for relevance feedback. For strategies which could not be used, such as Boolean search, searchers found alternative behaviors.

I. TERM STRATEGIES (TERM in Bates (1979))
   1. Identifying keywords
   2. Identifying synonyms
   3. Identifying controlled vocabulary
   4. Specifying phrases

II. DATABASE STRATEGIES (FILE STRUCTURE in Bates (1979))
   5. Understanding database

III. INTERACTION STRATEGIES
   6. Interaction with thesaurus
   7. Interaction with documents
   8. Magnitude feedback

IV. SEARCH STRATEGY (SEARCH FORMULATION in Bates (1979))
   9. Facet analysis
   10. Broad to narrow
   11. Boolean
   12. Specific search
   13. Restriction
   14. Iterative/interactive searching
   15. Structured query

Table 5. Categories of "usual" searching strategies

3.3 Characteristics of the search process

In addition to the time taken to do a search, which is constrained by the nature of the task that was given to the searchers, other important features of the search process, for our study, include: the use of relevance feedback; the nature of the queries which people construct; how the searchers interact with the documents and their surrogates; and, how their searching behaviors relate to their normal searching practices. Most of these characteristics are still under investigation, and many of them will be obtainable only through analysis of the interviews, and of the thinking-aloud protocols, which is now underway.

Here we discuss aspects of relevance feedback and query construction which are directly obtainable from the search logs, which seem to us to have some interesting implications. For a full description of the search process variables which we have investigated, and the values we found for them, see Section III of the Interactive System Description.

We begin by considering relevance feedback behavior. Only one searcher did not use relevance feedback at least once in at least one search iteration (Iteration is defined as all that takes place between one invocation of the Run Query button and the next, with the first iteration beginning with the searcher receiving the search topic, and the last ending with invoking the Exit button. Iterations are queries plus one). Of all of the queries in which relevance feedback was possible (699 out of a total of 846), relevance feedback was used in 491 (70.2%). The mean number of documents marked relevant per query was 4.9, adding, on average, 12.82 terms per relevance feedback query. There was no relation between the number of the iteration (how far in the search process an iteration was) and the number of documents marked relevant in that iteration. This finding may be an artifact of averaging across searches with different numbers of iterations, and needs further analysis.

As far as query terms are concerned, we considered two classes of such terms; those supplied by the searcher (*userwords)* and those supplied through relevance feedback. The mean number of userwords (which includes words in phrases as separate words) for all searches was 6.66; the mean number of userwords for the 93 correct first searches was 5.23. There was a slight trend toward increasing number of userwords with iteration (r=.09 p≤.01), but less than 1% of the variation in query length can be explained by iteration.

Thus, it appears that although our searchers engaged in substantial interaction with the rest of the IR system (9.46 iterations per search, 4.9 documents marked relevant per rf (relevance feedback) query), the nature of this interaction was, by and large not to increase the number of words in the query, but rather to change them. Although we have no measures, at the moment, that could tell us about any increase in search effectiveness through the search, it at least seems likely that the searchers continued to interact in order to improve their performance. It is also the case that the nature of the task that was set the searchers might have led to this form of interaction, since the focus was not on constructing an ever better query, but rather on finding (being able to see) more good documents.

A related interesting result is that there were significantly more userwords in queries that used relevance feedback than in queries that did not use relevance feedback (mean userwords per non-rf query = 6.1; Mean userwords per rf query = 7.22). Although this amounts to only one additional word per query, the difference is statistically significant (Mann-Whitney U' 4.01, p<.0001), and may even be considered substantively significant, given the relatively small number of words in the average query. Since all but one of our searchers used rf, this result seems unlikely to be due to general searcher characteristics. A possible explanation for this finding is that the increased number of highlighted terms in documents retrieved by rf searches either drew the searchers' attention to more terms in the texts which could be used for the query, or reminded them of terms which they could add. Again, it may be possible to get a better explanation of this finding from the thinking-aloud protocols.

In our interface, the display resulting from query submission is a list of the titles of the top ten retrieved documents, which is scrollable through the entire list. In order to read the text of a document, the searcher must explicitly request (by double-clicking on the title) the display. The mean number of times that the full text of a document was requested was 23.690 per search, with an Inter Quartile Range of 15-30. These data suggest that there was fairly substantial interaction with the full text of documents, despite relatively little experience with full-text retrieval systems (see Interactive System Description, II.1.i).

3.4 Searcher responses to relevance feedback and ranking

We have some informal (that is, anecdotal) results from the thinking aloud protocols and the post-search interviews concerning how our searchers used and understood relevance feedback, which may have interesting implications for implementation of this feature in interactive IR systems for end users. The comments we make here will need to be buttressed by more detailed and extensive analysis of our data, but as they stand they are at least suggestive of trends.

Many of the searchers commented that they wanted either explicit knowledge of the terms that had been added to their queries by relevance feedback, or, more strongly, that they wanted to have control over which terms, obtained through relevance feedback, would be added to their queries. The validity, or reasonableness of this desire is substantiated by recent results that indicate that even almost completely inexperienced searchers can effectively choose terms for queries offered through relevance feedback (Koenemann, 1995). This is in rather sharp contradiction to the commonly held belief that users in IR systems may not have sufficient knowledge of the workings of the system to make effective choices of this sort.

Another result that might have significant implications for relevance feedback implementation is that many of the searchers commented that they wished that they could indicate documents as not being good. This comment took several forms. Sometimes it was couched as a desire for a "NOT" operator, sometimes it was made explicitly by asking for the ability to mark a document "not relevant", sometimes it was expressed simply as "I wish I didn't have to see the same bad documents all the time". In the context of a system offering relevance feedback, these might all conflate to having the ability to use negative relevance judgments in the feedback process. Although this is commonly done in non-interactive environments, it is not at all common in interactive IR systems, usually on the grounds that "users don't want it".

The other consistent comment by the searchers who used relevance feedback successfully (or rather, thought that they did so), was that using relevance feedback for query expansion or modification was *easier* than doing it oneself. Finding good terms for searching is an arduous process, and relevance feedback seemed to ease this burden substantially, for a substantial number of our searchers, both by supplying terms, and by suggesting them.

We wish also to note the importance of distinguishing between "saved" documents and "relevant" documents, at least in our system, for this task. During the "thinking-aloud", a number of searchers made comments that explicitly separated these functions, for instance, noting that a document was "good", and should be saved, but that it would not be used for relevance feedback, because that would only "get more like it", or because it "didn't have good words in it". Conversely, a document might be marked for relevance feedback, but not saved, because it "has good words in it", but is "not quite on the topic". Such comments, and the general idea of distinguishing between the relevance feedback function and the saving of documents, indicate that searchers with no (or almost no) experience of relevance feedback can rapidly develop sophisticated models of its operation and manipulation.

Our searchers were close to unanimous in their approval of ranked output, although there was also evident, in the thinking-aloud protocols, some confusion between the length of the ranked list (set at a maximum of 150) and an unranked set of size 150. In general, our searchers appear to have made effective use of the ranking mechanism, primarily by working to move good documents to the top of the list (again, according to their comments while thinking-aloud). The positive comments on ranking had usually to do with "getting good stuff at the top", and "not having to look through the whole list". However, seeing documents that had been previously identified as not good continue to appear fairly high in subsequent ranked lists was disturbing to many searchers.

4. Conclusions

Although our results are still preliminary, and also rather incomplete, we feel that it is possible to make some tentative conclusions based on them. The most obvious one, of course, is that it appears that, for this explicit task, support for interactive searching is valuable. Although we have no comparative data with automatic systems on the adhoc task, our (admittedly rather impoverished) baseline search is not an unreasonable approximation of how an automatic system without too many frills would perform, given the sorts of queries that we have faced. And our interactive searchers appear to have done rather better than the baseline in this task.

It is probably also safe to conclude, on the basis of our data to date, that people who have had little or no experience of relevance feedback and ranking can, with relatively little training, learn to use these features quite effectively. Although there are certainly instances in our data of people who had some difficulty in understanding and using both relevance feedback and ranking, it seems that many such problems were attributable to forgetting about the features entirely, or to forgetting just what they did (especially for relevance feedback).

Our findings so far, with respect to these features, have some interesting implications for how such features might be implemented in end-user IR systems. For instance, it seems clear that we should at least try to give searchers more knowledge of what happens in relevance feedback, and probably to give them more control over just what is actually done (the latter suggestion cries for further empirical studies). Furthermore, it seems that some mechanism should be provided for explicitly moving non-relevant documents down in the lists of retrieved documents, if not removing them from the lists entirely. Negative relevance feedback seems at least a reasonable candidate for such a mechanism. And it appears that some method of explicit explanation of the reasons for document rankings will be necessary, in order both to help people interpret the rankings, and to support them in their interactions with the database.

Although others have also found little relationship between individual differences of searchers and search outcomes (e.g. Saracevic & Kantor, 1988), we were nevertheless somewhat surprised that we found no significant relationships in this rather different situation. This finding is still rather tentative, since we have some other characteristics, such as mental models of IR, yet to investigate. But, it, combined with the rather clear differences in performance indicated in Table 4, suggests that we need further to investigate whether features supporting interaction in IR systems can lead to reducing differences in performance between searchers.

This brings us to our final comments, which concern the nature of evaluation of performance in interactive IR systems. It seems to us reasonably clear that the task that has been set the interactive track this year, and the primary method for evaluating that task, may not be very good for evaluating and comparing different interactive IR systems. Precision and recall on the set of saved items has obvious problems, some of which we noted above. Indeed, evaluating on the basis of saved items, in response to someone else's information problem, may have turned out to be an experiment in consistency of relevance judgments between searchers and relevance assessors, rather than an evaluation of system performance. For instance, for topic 243, of the 46 documents explicitly saved by the four searchers, only 11 were judged relevant by the assessors. Furthermore, the task that was set, retrieving as many good documents as possible in some time period, unfortunately does not relate well to many of the information problems represented by the adhoc topics. These problems often suggested that *one* document, were it to answer a specific question, would be far preferable to many documents, each of which, while eminently topically relevant, was completely redundant with the others.

Although issues of this sort have been raised before in IR evaluation, it seems that the interactive context makes them especially important. Indeed, our searchers, when asked in their pre-search interview how they would decide when a search was finished, rarely answered that this would be when they felt that they had found all of the good documents in the database. On the other hand, as Su (1992) has found, users of interactive IR systems seem not to be too worried about precision, either. All of this suggests, to us, that if we are going to be serious about evaluating effectiveness in interactive IR, we need to develop some new kinds of task environments in which to do the evaluation (implying some new methodologies, as well), and some new performance measures. The former seems especially to require that we look at information problem or tasks that are "real" to the searchers whom we are studying; the latter, that we develop measures or descriptions based upon the search process itself, and upon the task which has led the searchers to engage in the IR situation. It is our hope that the experience of this year's interactive track at TREC will provide a basis for such a new evaluation paradigm.

Acknowledgments

Aravindan Veerasamy, from Georgia Tech, spent a summer working with us on this project. We are indebted to him for his participation, including study design, data collection, talking about things, being a good guy, and lots of stellar programming. The Center for Intelligent Information Retrieval at the University of Massachusetts provided us with invaluable support in our use of INQUERY. And we owe our greatest thanks to the wonderful people who volunteered to be the searchers in this study.

References

Bates, M.J. (1979) Infoımation search tactics. *Journal of the American Society for Information Science*, 30: 205-213.

Callan, J.P., Croft, W.B. & Harding, S.M. (1992) The INQUERY retrieval system. In: *DEXA 3*. Proceedings of the Third International Conference on Database and Expert Systems Applications. Berlin: Springer Verlag, 83-87.

Haines, D. & Croft, W.B. (1993) Relevance feedback and inference networks. In: R. Korfhage, E. Rasmussen & P. Willet, eds. *SIGIR '93*. Proceedings of the Sixteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. New York: ACM, 2-11.

Hewett, T.T. & Scott, S. (1987) The use of thinking-out-loud and protocol analysis in development of a process model of interactive database searching. In: H.-J. Bullinger & B. Shackel, eds. *INTERACT '87*. Amsterdam: Elsevier & IFIP, 51-56.

Koenemann, J. (1995) Relevance feedback: Usage, usability, utility (abstract of the poster presentation). In: E.A. Fox, P. Ingwersen & R. Fidel, eds. SIGIR '95. Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. New York: ACM, 372.

Koenemann, J., Quatrain, R., Cool, C. & Belkin, N.J. (1995) New tools and old habits: The interactive searching behavior of expert online searchers using INQUERY. In: D. Harman, ed. *TREC-3*. Proceedings of the Third Text REtrieval Conference. Washington D.C.: GPO, 145-177.

Van Rijsbergen, C.J. (1979) *Information rętrieval*, 2nd ed. London: Butterworths.

Saracevic, T. & Kantor, P. (1988) A study of information seeking and retrieving. III. Searchers, searching, and overlap. J*ournal of the American Society for Information Science*, 39 (3), 197-216.

Su, L.T. (1992) Evaluation measures for interactive information retrieval. *Information Processing and Management*, 28 (4), 503-516.

Appendix A. Experimental Materials

RUTGERS TREC 4 INTERACTIVE SEARCHING EXPERIMENT

SEARCHER QUESTIONNAIRE

1. Please indicate below all the college/university degrees you have (or expect):

_______________ ____
Degree Major

_______________ ____
Degree Major

_______________ ____
Degree Major

2. How much experience have you had doing online searching with the following kinds of systems? Please answer by circling a number on the response scale.
[here we give a five-point scale, from None (1) to A great deal (5) for each of the following kinds of systems]
Computerized Library Catalogs
CD ROM Systems, e.g. Infotrac
Commercial online systems, e.g. Dialog, Lexis, BRS Afterdark
World Wide Web Browsers, e.g. Mosaic, Netscape
Other systems, please specify.

3. Overall, how many years have you been doing online searching? ___________ years

4. Do you do online searches for:
______ Yourself only
______ Others, e.g. as an intermediary
______ Yourself and others

5. How much experience have you had searching **full-text** databases? [five-point scale, None to A great deal]

6. How much experience have you had searching in **ranked-output** information retrieval systems? [as above]

7. How much experience have you had searching in information retrieval systems which offer **automatic relevance feedback**? [as above]

8. How often do you use a computer in your workplace, school or home?
_____Daily _____Once or twice a month _____Never
_____Once or twice a week _____Less than monthly

9. How much experience have you had using a mouse-based interface? [five point scale from None to A great deal]

10. What is your age?
_____Under 21 _____21-30 _____31-40 _____41-50 _____51-60 _____over 60

11. Are you: _____Female _____Male

TREC 4 PRE-SEARCH INTERVIEW

In order to understand the different searching experiences of our participants, we would like you to answer a few questions about the methods you typically use when you do online searching. When you answer these questions, please try to give as much detail as you can. You may use the worksheet for any notes you wish to make. (Hand worksheet to searcher)

Imagine that you are interested in finding information on the following topic and that you have available to you a computerized database of newspaper articles which you can search. [This paragraph, and

Topic: Identify alternative sources of energy for automobiles. Include additives to gasoline that either decrease pollution or reduce oil consumption.

[The interviewer reads the topic. The problem description and topic are printed at the top of the worksheet.]

1. How would you go about putting together a search on this topic? What would be your overall approach? Try to describe what you would do first, and then the steps you would go through after that.

2. On the worksheet, please write down what you think your initial search statement, or query, to the system would be. Write the query just as you would enter it into the system you normally search. Do this next to #2 on the worksheet.

3. After you've run the query, what would you do to determine if it had retrieved any good documents?

4. What do you think you would do if your query didn't retrieve any good documents?

5. How would you decide that your search is finished?

RUTGERS TREC 4 INTERACTIVE SEARCHING EXPERIMENT

SEARCH EVALUATION FORM

TOPIC NUMBER__________

1. How familiar are you with this topic? [five points, Not at all (1) Somewhat (3) Very much (5)]

2. How easy was it to do this search? [five points, Not easy (1) Somewhat (3) Very easy (5)]

3. How satisfied are you with your search results? [five points, Not at all (1) Somewhat (3) Completely (5)]

4. Do you feel you had enough time to do an effective search? [five points, Far too little (1) Just enough (3) Too much (5)]

TREC 4 POST-SEARCH INTERVIEW

At the beginning of our session today you talked about your typical online searching methods. Now think about the two searches you just conducted with the RU-INQUERY retrieval system.

1. Would you say that you followed your routine searching methods completely, somewhat or not at all when you did these searches?
   Completely ____ Somewhat ____ Not at all____ (go to Q3)

2. Which of your routine searching methods did you use in these searches?

3. Which methods did you NOT use in these searches? (For each method mentioned, ask:) Why did you not use this method? What did you do instead?

4. Did you use or try to use the Relevance Feedback feature in either of your searches?
   a. Did not try to use RF ____
      Ask: Why not?
   b. Tried RF unsuccessfully ____
      Ask: What happened?
   c. Used RF ____
      Ask: In what ways was this feature helpful or not helpful?

5. In what ways was it helpful or not helpful to have Ranked Output in your searches?

6. In what ways was it helpful or not helpful to have Full Text?

7. Finally, are there any comments that you have on your experience with RU-INQUERY? Features that you liked or didn't like, or features that you wish you'd had?

Interactive System Description for Rutgers TREC-4 Interactive Study (runit1)

I. System description

I.1 Screen dump of a typical screen (Figure 1).

I.2 Usable features of the interface

I.2.a General

Our retrieval engine is INQUERY 2.1p3, distributed by the Center for Intelligent Information Retrieval, University of Massachusetts. We reduced the functionality of the query formulation and interface features for our experiment, while basing the interface and general features on the distributed system. The interface consists, from top to bottom, of: a ine with buttons for query control and manipulation; a query formulation window; an information line; a results summary window; a text reading window; and a keyword finding button.

I.2.b Query formulation features

A query is entered in the **query formulation window** at the top of the screen. Queries are unstructured, with no operators except for the "hyphen", which indicates that the words connected by the hyphen must appear adjacent to one another, in the stipulated order. Common query phrases, such as 'give me information on' are automatically eliminated by the INQUERY query stop-phrase routines. Queries can extend over multiple lines, and the window is scrollable.

Run Query runs the current query in the query formulation window (modified according to any changes due to relevance feedback). If one or more words in the query do not occur in the database, a pop-up window gives this message, and asks the user to check the spelling of the word (displayed) or to remove it from the query. **Save Query** opens a window which allows the current query to be saved to a specific name. **Load Query** opens a window which allows any saved query to be put into the query formulation window (if there is already a query there, the loaded query is inserted at the cursor). **Clear Query** removes the current query from the query formulation window. **Clear Rel Feedback** removes the indication of relevance from all documents so marked, and reverts the query to that in the query formulation window only. **Exit** ends the interaction by closing the interface.

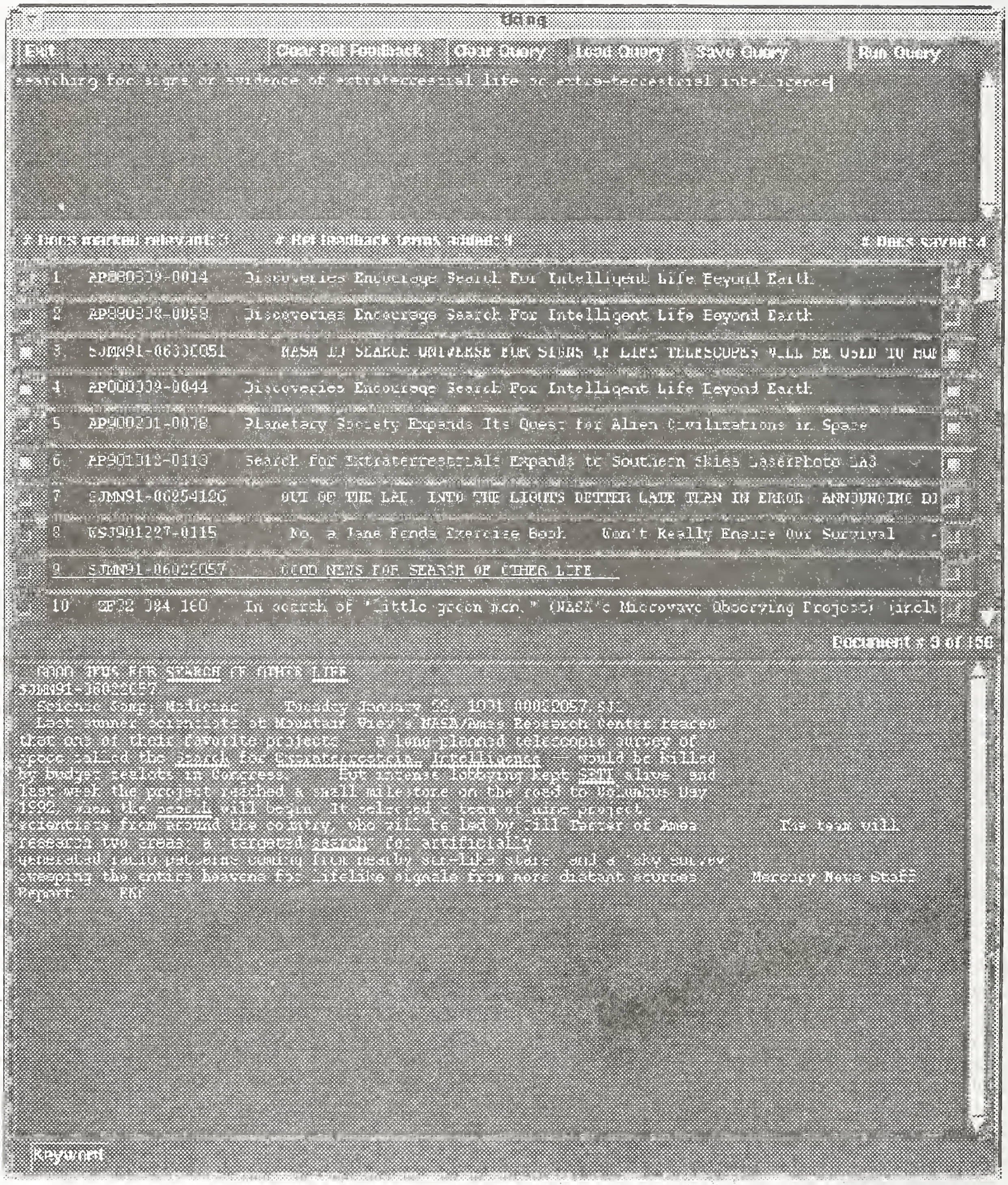
I.2.c. Information features

In the information lines below the query formulation window, information is displayed, from left to right, about the number of documents that have been marked relevant, to be used for relevance feedback; the number of terms that have been added to the query due to relevance feedback (this information is provided as each document is marked); and the number of documents that have been marked to be saved.

I.2.d. Search results

Below the information line is a **"summary" window**, which displays, from left to right: the rank; the document id; and, the title of the document, for ten documents. This list is scrollable, ten document titles at a time.

On the left of each title display is a box which can be clicked on to mark an item as **relevant**, for the purpose of automatic relevance feedback. On the right of each title display is a box which can be clicked on to indicate that an item is to be "**saved**" to a set of saved documents. Both relevant and save boxes are toggles which change from blue to yellow when invoked, and back to blue when cleared.



```
searching for signs or evidence of extraterrestrial life or extra-terrestrial intelligence

# Docs marked relevant: 3     # Rel feedback terms added: 9                    # Docs saved: 4

 1  AP880309-0014   Discoveries Encourage Search For Intelligent Life Beyond Earth
 2  AP880308-0058   Discoveries Encourage Search For Intelligent Life Beyond Earth
 3  SJMN91-06330051     NASA TO SEARCH UNIVERSE FOR SIGNS OF LIFE TELESCOPES WILL BE USED TO HUN
 4  AP000309-0044   Discoveries Encourage Search For Intelligent Life Beyond Earth
 5  AP900201-0078   Planetary Society Expands Its Quest for Alien Civilizations in Space
 6  AP901012-0113   Search for Extraterrestrials Expands to Southern Skies LaserPhoto LA3
 7  SJMN91-06254126     OUT OF THE LAB, INTO THE LIGHTS BETTER LATE THAN IN ERROR  ANNOUNCING DI
 8  WSJ901227-0115      No, a Jane Fonda Exercise Book    Won't Really Ensure Our Survival   -
 9  SJMN91-06022057     GOOD NEWS FOR SEARCH OF OTHER LIFE
10  ZF32 384 160    In search of "little green men." (NASA's Microwave Observing Project) (inclu

                                                                         Document # 9 of 150

 GOOD NEWS FOR SEARCH OF OTHER LIFE
SJMN91-06022057
 Science Comp; Medicine     Tuesday January 22, 1991 00022057 SJ
 Last summer scientists at Mountain View's NASA/Ames Research Center feared
that one of their favorite projects -- a long-planned telescopic survey of
space called the Search for Extraterrestrial Intelligence -- would be killed
by budget cutters in Congress.     But intense lobbying kept SETI alive, and
last week the project reached a small milestone on the road to Columbus Day
1992, when the search will begin. It selected a team of nine project
scientists from around the country, who will be led by Jill Tarter of Ames.        The team will
research two areas: a targeted search for artificially
generated radio patterns coming from nearby sun-like stars, and a sky survey
sweeping the entire heavens for lifelike signals from more distant sources.      Mercury News Staff
Report.    END
```

Figure 1. Screen dump of "typical" RU-INQUERY interface.

Double-clicking on a title brings up the text of that document in the **document window** at the bottom of the screen (the document window is empty until a document is explicitly selected for display). The title of the document which is being displayed is underlined in the summary window. The rank of the displayed document is indicated in an information line between the summary window and the document window, with the total number of retrieved documents (set to a default maximum of 150 documents in the retrieved list). The text of the document is scrollable, either by scrollbar on the right, or by clicking on the Keyword button at the lower left of the screen, which scrolls the text display to the next occurrence of any query word. All query words are underlined and highlighted, including those added by relevance feedback.

I.3 Style of Interface

Graphical User Interface (GUI)

II. Experimental Conditions

II.1 Searcher Characteristics

a. **Number of searchers in experiment**: 51 (includes one searcher whose results were not used because of a system failure)

b. **Number of searchers per topic**: 4

c. **Age group of searchers:**

Age	n(searchers)
Under 21	0
21-30	10
31-40	18
41-50	16
51-60	5
Over 60	2

d. **Gender of searchers**

Female: 33 **Male:** 18

e. **Educational level of searchers** (All searchers had at least a Bachelor's degree, many had multiple advanced degrees):

Degree	n(searchers)
M.A.	24
MLS	39
Ph.D. (expected)	9
Ph.D. (earned)	5
JD	3

f. **Purpose of searching**:

For oneself: 20

For oneself & others: 31

g. **Frequency of computer use**:

Once or twice a week: 9

Daily: 42

h. **Experience with mouse-based interface**: (Scale 1-5, where 5 is "A great deal". No searchers reported 1 "None".)

MEAN: 4.402

i. **IR searching experience** (all on 1-5 scales, with 1=None, 3=Some, 5=A great deal)

With Computerized Library Catalogs

Value	n(searchers)
1	0
2	3
3	11
3.5	1
4	20
5	16

MEAN: 3.971

With CD-ROM Systems

Value	n(searchers)
1	2
2	12
3	22
4	10
5	5

MEAN: 3.078

With Commercial Online Systems

Value	n(searchers)
1	4
2	11
3	18
4	11
5	7

MEAN: 3.118

With WWW Browsers

Value	n(searchers)
1	6
2	15
3	11
4	11
5	8

MEAN: 3.0

With Other Systems

Value	n(searchers)
1	40 (36 no response)
2	1
3	5
4	1
5	4

MEAN: 1.588

Years of Searching Experience:

MEAN: 5.517 MEDIAN: 3.500

MIN: 0 MAX: 25 SD: 5.754

With Full-Text Databases

Value	n(searchers)
1	10
2	15
3	17
4	5
5	4

MEAN: 2.569

With Ranked-Output Systems

Value	n(searchers)
1	25
2	14
3	10
4	1
5	1
MEAN: 1.804	

With Relevance Feedback Systems

Value	n(searchers)
1	37 (1 non-response)
2	6
3	6
4	1
5	1
MEAN: 1.490	

II.2 Task description (Below are the verbatim instructions given to the searchers on the search form. Searchers were instructed to "believe that this is a topic that you, yourself, are interested in".)

Your task is to find as many documents as you can which address the information problem given below, but without too much rubbish. You should complete the task in around 30 minutes or less.

[Here, the topic of the search was displayed]

In the RU-INQUERY system, for the purposes of this task, you may find and save documents in two ways. One is to mark the documents you think are good with the ***Save Document*** *buttons; the other is to make a query which ranks a lot of good documents at the top of the list, and to* ***Save*** *this* ***Query****. If you do the latter, please name the query that you save* ***final****. You may, if you want, use both methods in combination.*

Please remember that the database in which you are searching has documents in it which were published between 1988 and 1992, so you will not be able to find documents referring to current events.

When you have completed your task (or when your time is up), and saved all the documents you like, and/or a final query, please click on the ***Exit*** *button at the top left of the RU-INQUERY interface.*

Please remember to ***think aloud*** *while you are doing the search (including while you are reading the description of the information problem).*

Thanks, and have fun!

II.3 Training

II.3.a Description of training process.

After having been interviewed about their normal searching behavior, searchers were seated at the computer on which they would be searching, given a loose-leaf folder with a printed tutorial for the RU-INQUERY system, and were instructed to follow the tutorial completely, by interacting with the system according to the directions in the tutorial. Screen dumps of screens they would encounter during the tutorial were also provided. The entire tutorial was based around a search on the topic of "documents reporting on the possibility of, and search for extra-terrestrial life and/or intelligence", one of the TREC-4 adhoc topics not in the set of interactive topics, and was conducted on the TREC 4 adhoc database.

The tutorial was divided into three sections. Section one was concerned with the mechanics of the interface, with using the features of RU-INQUERY for constructing and submitting queries, and with interpreting the system output. Ranked output was explained in the tutorial as follows:

The RU-INQUERY system ranks all of the documents in the database according to how well it believes each document will satisfy your information problem; that is, the document's *relevance**. It does this in a very simple way. The rank of a document is based on:*

- *how many of the words in your query appear in the document;*
- *how often these words appear in the document;*
- *how common the words that match are in the document collection as a whole (the more common they are, the less important);*
- *the length of the document.*

RU-INQUERY combines all of these factors into a single score that is then used to rank all of the documents, with the document having the highest score being ranked 1. Notice that this means that documents do not have to match your query exactly in order to be retrieved and ranked, as they do in many other information retrieval systems. In general, documents which contain at least one word that is in your query (except for very common words like and, the, of, and so on, on the *stop list**, which are ignored by the system), will be ranked.*

Section two of the tutorial explained the use of the Save Query and Load Query features, and the use of the Relevance Feedback and Save Document buttons. Relevance feedback was explained in the tutorial as follows:

In RU-INQUERY, you can tell the system that a document is relevant, or good. If you do this, the system will automatically modify your original query, the next time you run it, in order to try to find more documents like the one(s) which you marked relevant. It does this modification by adding new terms to the query from documents which you have marked relevant, and by making some terms which appear often in relevant documents more important in the query. This is called *relevance feedback**.*

Relevance feedback typically has three effects: documents which you have marked relevant will be moved higher up the ranking; new documents which are similar to those which you have marked relevant will be added to the list of retrieved documents; and documents which are similar to those you marked relevant and were already in the retrieved list, will be moved higher up the ranking.

In Section two, the interactions between Clear Query, Clear Rel Feedback and Save Query were demonstrated, as was the difference between the Save Document and Relevant Document features.

Section 3 was a practice search, using the same problem description as for the task (see II.2, above), but limited to fifteen minutes (noting that they would have 30 minutes in the real searches). The practice search was on one of the TREC 4 adhoc topics not in the interactive set (224), and was introduced as follows:

For this study, you will be asked to do two searches for documents which will be useful in addressing real information problems. To give you some more experience with using RU-INQUERY, and with the task you will be doing, we would like you to do a practice search of this type.

Imagine that you are interested in the following information problem:

What can be done to lower blood pressure. for people diagnosed with high blood pressure? Include benefits and side effects.

Your task is to find as many documents as you can which address this problem, but without too much rubbish. You should complete the task in around 15 minutes or less (in the real searches, you will have up to 30 minutes).

II.3.b Time for training

The training times (in minutes) are for Sections one and two only. All participants used their allotted fifteen minutes for the practice search, which can be added for a total tutorial time.

MEAN: 39.98
MIN: 20 MAX: 65

III. Search Process

1. Clock time (i.e. real, elapsed time) per search, from the time the searcher was given the topic, until the searcher exited the system given in minutes. Note that searchers were instructed that they had 30 minutes in which to do the search.

MEAN: 26.850 MEDIAN: 29
MIN: 11 MAX: 35 SD: 5.842

2. Number of documents "viewed" during a search. We have three categories of "viewing": **all titles**, defined as the total number of document titles displayed during the search; **unique titles**, defined as the number of unique titles displayed during the search; **Text**, defined as the number of documents whose text the user displayed during the search.

Viewing Behavior				
Viewing Type	Mean	Median	SD	Range
All Titles	321.470	300	181.872	50 - 900
Unique Titles	139.780	122.500	82.968	22 - 391
Text	23.690	22.000	11.572	5 - 66

3. Number of iterations per search. An iteration is all that occurs between one invocation of "Run Query" and the next, with the first iteration beginning when the searcher is handed the topic, and the last ending when the searcher "Exits" the system. Note that this number corresponds to the number of queries submitted to the system, plus one (for the last iteration, ending with Exit).

MEAN: 9.46 MEDIAN: 8.5
MIN: 2 MAX: 28 SD: 5.231

4. Query characteristics

a. **Total queries for all searches**: 846

b. **Total correct queries that retrieved documents**: 799

c. **Queries that retrieved no documents**: 5

d. **Queries that contained words not in database, or which were syntactically incorrect**: 42

5. Use of relevance feedback

There were 746 non-first queries total, of which 47 followed queries that retrieved no documents or which had errors, leaving 699 queries for which searchers had the opportunity to use relevance feedback.

a. **Total usage**: 491 queries using relevance feedback (70.2%)

b. **Number of relevant documents per query** (of the 491 queries)

MEAN: 4.9 MEDIAN: 3.0
MIN: 1 MAX: 45 SD: 4.96

c. **Number of terms added by relevance feedback per query**

MEAN: 12.82 MEDIAN: 9
MIN: 5 MAX: 93 SD: 9.9

6. Numbers of terms used in queries

a. **Userwords per query** (meaning words entered by the searcher in the query formulation window, and counting hyphenated terms as the sum of the words in the term)

MEAN: 6.66 MEDIAN: 5
MIN: 1 MAX: 162 SD: 8.6

b. **Total query length** (meaning user words plus words added by relevance feedback)

MEAN: 14.06 MEDIAN: 11
MIN: 1 MAX: 162 SD: 13.75

c. **Total query length for non-feedback queries** (n=330)

MEAN: 5.86 MEDIAN: 4
MIN: 1 MAX: 162 SD: 9.46

d. **Total query length for feedback queries** (n=474)

MEAN: 19.78 MEDIAN: 16
MIN: 6 MAX: 113 SD: 13.38

e. **Number of userwords per feedback query**

MEAN: 7.22 MEDIAN: 5
MIN: 2 MAX: 76 SD: 7.96

7. Use of system features

Apart from use of relevance feedback and viewing of full texts, as discussed above, these data are not yet computed.

IV Transcript for Topic 236 (Searcher 047)

What follows is the log of the interaction for topic 236, searcher 047. Given are the elapsed time, a description of the searcher's actions, system responses, and extracts from the searcher's verbal protocol. The searcher comments, enclosed in quotation marks below, have been abbreviated for this transcript. The full text of viewed documents is reproduced for short documents, for long documents only the approximate amount that fit on the screen and was visible without scrolling is given.

```
00:00      USER: Start search; reads topic description

01:45      USER: "Current laws of the sea. Do they mean like if you're in a ship or do they mean like how nature operates in the sea?  Hmm, that's a good one." (rereads
instructions)

03:45      USER: "I think I'm gonna start with laws of the sea and see just what that brings up for me."

04:01      USER: "Something I miss in this database are the descriptor fields. I'm sort of acting  in a Boolean fashion. I don't feel like I'm fully taking advantage of what this
system offers."

04:01      SYS:        Submitted Query:
                       laws of the sea
                       and 0 marked documents

04:03      SYS:        Result of Query Evaluation: 150 documents retrieved
04:05      SYS:        Display of Titles of Retrieved Documents:

                        NR NS 1.  AP881101-0176   U.S. Votes Against Law Of Sea Convention
                        NR NS 2.  AP901113-0077   Monterey Looks for Federal Law to Help Evict Sea Lions
                        NR NS 3.  AP881031-0251  AGENCIES AND RADIO OUT From AP Newsfeatures
                        NR NS 4.  AP900622-0055   Administrative Law Judge Fines Fishermen in Turtle Case
                        NR NS 5.  WSJ900514-0065       LAW    ----    By Milo Geyelin
                        NR NS 6.  FR88105-0063
                        NR NS 7.  AP900901-0010   Death Threats Over Marine Mammals
                        NR NS 8.  AP900512-0030   Sea Lions Still Making Pier 39 Their Home LaserPhoto FX1
                        NR NS 9.   AP901220-0027    Kuwaitis Paid PR Firm $5.64 Million In 3 Months Since Invasion 19 Texans Fined For Altering Turtle Excluder Devices
President               to Consider Oil Import Tax
                        NR NS 10.  FR88616-0064

04:40      USER: "Ok, let's look at number 1."

04:41      USER: Display Full Text of Document ranked 1

04:42      SYS:
                       U.S. Votes Against Law Of Sea Convention
AP881101-0176
UNITED NATIONS (AP)

   The General Assembly on Tuesday voted in
favor of the Convention on the Law of the Sea and for the sixth
consecutive year the United States cast its ballot against the
resolution.
   Washington has said the convention would infringe on U.S. rights
to exploit seabed resources.
   Turkey joined the United States in casting its vote against the
resolution that urged all nations to ratify the 1982 convention.
Turkey is in a dispute with Greece over exploration rights in the
Aegean Sea.
   The resolution was approved by a vote of 135-2 with six
abstentions. The rest were absent.
   Sixty nations in the 159-member assembly must ratify the
convention before it would become international law. So far 35
countries have signed the pact.
   It would establish an agency to govern uses of the sea in
international waters, traditionally an unrestricted area.
   The agency would deal with commercial disputes, protectio

[remaining text deleted...]
05:17      USER: "Great. I'm gonna mark this document as relevant, and rerun my query."

05:18      USER: Mark as Relevant document ranked 1 with document ID AP881101-0176
05:23      SYS:        Submitted Query:
                       laws of the sea
                       and 1 marked documents
05:26      SYS:        Result of Query Evaluation: 150 documents retrieved
05:28      SYS:        Display of Titles of Retrieved Documents:

                         R NS 1.  AP881101-0176    U.S. Votes Against Law Of Sea Convention
                        NR NS 2.  AP900201-0221    Dead Sea Was an Active Trade Route as Far Back as 2,300 Years
                        NR NS 3.  FR881027-0027
                        NR NS 4.  AP900724-0203    Part III: The Catastrophe A Great Sea Is Dying; `Hundreds Will Die With It' LaserPhoto NY307 of July 24; Graphics
                        NR NS 5.  SJMN91-06324011     SOVIET LAKE'S DEATH MIRRORED IN CALIFORNIA
                        NR NS 6.  AP880707-0139    Safety Questions Over North Sea Oil Rig
                        NR NS 7.  FR881024-0194
                        NR NS 8.  AP880405-0032    Oil Men Tend Garden On North Sea Oil Rig
                        NR NS 9.  AP880711-0049    Three More Bodies Found In North Sea Platform Disaster With PM-Oil City, Bjt
                        NR NS 10.  PT3-05154561      Automated all-weather cargo transfer system

06:03       USER: "Well, my number one document is still number one."
06:05       USER: Display Full Text of Document ranked 2
06:06       SYS:
                        Dead Sea Was an Active Trade Route as Far Back as 2,300 Years
AP900201-0221
QUMRAN, Occupied West Bank (AP)
By NICOLAS B. TATRO Associated Press Writer
   Researchers say the Dead Sea
was a lively commercial center as far back as 2,300 years ago, not
anywhere near like it is today with few boats and only sparse
settlement along the shores.
   The research may also throw new light on the Essenes who lived
in the area, casting doubt on the accepted theory that they were
isolated hermits.
   Arieh Nissenbaum, a geo-chemist, said three stone anchors and
mooring ropes found recently in the Dead Sea were the ``first
datable evidence'' to support the theory that the Dead Sea was an
active place for commerce in the 4th century B.C.
   He said Carbon 14 testing at the Weizmann Institute of Science
in Rehovot on well-preserved fiber rope fragments left open the
possibilty that the anchors were from ships sunk in the first
Middle East oil war.
   ``The Dead Sea was not so dead. This is a m

[remaining text deleted...]

06:16       USER: Scroll Full Text of Document, lines   27 ff now visible
06:19       USER: Scroll Full Text of Document, lines   54 ff now visible

06:24       USER: "It doesn't look good. Bad document for my number two ranked document. Let's look at number three."
06:26       USER: Display Full Text of Document ranked 3

06:29       SYS:

FR881027-0027

Federal Register   / Vol. 53, No. 208 / Thursday, October 27, 1988
/ Proposed Rules
 DEPARTMENT OF TRANSPORTATION Coast Guard 33 CFR Parts 151, 155 and 158 46 CFR Part 25  [CGD 88-002] RIN 2115-AC89 Regulations Implementing the Pollution
Prevention Requirements of
Annex V of MARPOL 73/78 AGENCY:  Coast Guard, DOT.
 ACTION:  Notice of proposed rulemaking.
 SUMMARY:  The Coast Guard proposes rules to implement the requirements
of the ``Act to Prevent Pollution from Ships,'' as recently amended by
Congress. These rules will implement Annex V of the International Convention
for the Prevention of Pollution by Ships, 1973 entitled ``Regulations for
the Prevention of Pollution by Garbage by Ships.'' These regulations would
apply to marine craft of any size or type, including commercially operated
ships and fishing vessels, uninspected vessels, recreational boats, oil
rigs and platforms. These proposed rules also specify that ports and terminals,
including recreational boat

[remaining text deleted...]
06:37       USER: "Hm, Federal Register. I'm really not sure what's in this database."
06:39       USER: Scroll Full Text of Document, lines   26 ff now visible
06:47       USER: Scroll Full Text of Document, lines   53 ff now visible
06:53       USER: Scroll Full Text of Document, lines   78 ff now visible

06:54       USER: "I'm not seeing terms highlighted here. Oh, the keyword button. Take me to the next keyword."
06:56       USER: Scroll Full Text of Document to next matched keyword (14 times)

07:26       USER:  "I just thought of something. Law of sea convention. What about if I use hyphens to say look for it in this order. There's no descriptors in here. I'll try it."
07:50       USER:       Clear All Relevance Markings
07:56       USER:       Run Query
07:56       SYS:        Submitted Query:
                        law-of-sea
                        and 1 marked documents
07:57       SYS:        Result of Query Evaluation: 1 documents retrieved
07:57       SYS:        Display of Titles of Retrieved Documents:

                        NR NS 1.  AP881101-0176    U.S. Votes Against Law Of Sea Convention

07:59       USER: "Hm, there's one article."
07:59       USER: Rereads topic description.

08:10       USER: "We know that the number one ranked document is relevant. I'm rerunning my original query formulation with that document marked relevant, and I will look
at these    subsequent records."
08:37       USER: Mark as Relevant document ranked 1 with document ID AP881101-0176
08:40       USER: Run Query
08:41       SYS:        Submitted Query:
                        laws of the sea
                        and 2 marked documents
08:43       SYS:        Result of Query Evaluation: 150 documents retrieved
08:44       SYS:        Display of Titles of Retrieved Documents:

                         R NS 1.  AP881101-0176    U.S. Votes Against Law Of Sea Convention
                        NR NS 2.  AP900201-0221    Dead Sea Was an Active Trade Route as Far Back as 2,300 Years
                        NR NS 3.  FR881027-0027
                        NR NS 4.  AP900724-0203    Part III: The Catastrophe A Great Sea Is Dying; `Hundreds Will Die With It' LaserPhoto NY307 of July 24; Graphics
                        SOVNEWSO3, SOVNEWSOL
                        NR NS 5.  SJMN91-06324011     SOVIET LAKE'S DEATH MIRRORED IN CALIFORNIA
                        NR NS 6.  AP880707-0139    Safety Questions Over North Sea Oil Rig
                        NR NS 7.  FR881024-0194
                        NR NS 8.  AP880405-0032    Oil Men Tend Garden On North Sea Oil Rig
                        NR NS 9.  AP880711-0049    Three More Bodies Found In North Sea Platform Disaster With PM-Oil City, Bjt
                        NR NS 10.  PT3-05154561      Automated all-weather cargo transfer system

08:52       USER: Display Full Text of Document ranked 2
```

```
08:52       SYS:
                        Dead Sea Was an Active Trade Route as Far Back as 2,300 Years
AP900201-0221
QUMRAN, Occupied West Bank (AP)
By NICOLAS B. TATRO Associated Press Writer
   Researchers say the Dead Sea
was a lively commercial center as far back as 2,300 years ago, not
anywhere near like it is today with few boats and only sparse
settlement along the shores.
   The research may also throw new light on the Essenes who lived
in the area, casting doubt on the accepted theory that they were
isolated hermits.
   Arieh Nissenbaum, a geo-chemist, said three stone anchors and
mooring ropes found recently in the Dead Sea were the ``first
datable evidence'' to support the theory that the Dead Sea was an
active place for commerce in the 4th century B.C.
   He said Carbon 14 testing at the Weizmann Institute of Science
in Rehovot on well-preserved fiber rope fragments left open the
possibilty that the anchors were from ships sunk in the first
Middle East oil war.
   ``The Dead Sea was not so dead. This is a m

[remaining text deleted...]
08:57       USER: "Ok, this dead sea thing in the number two position is no good."
09:00       USER: Display Full Text of Document ranked 3

09:02       SYS:

FR881027-0027

Federal Register   / Vol. 53, No. 208 / Thursday, October 27, 1988
/ Proposed Rules
 DEPARTMENT OF TRANSPORTATION Coast Guard 33 CFR Parts 151, 155 and 158 46 CFR Part 25  [CGD 88-002] RIN 2115-AC89 Regulations Implementing the Pollution
Prevention Requirements of
Annex V of MARPOL 73/78 AGENCY:  Coast Guard, DOT.
 ACTION:  Notice of proposed rulemaking.
 SUMMARY:  The Coast Guard proposes rules to implement the requirements
of the ``Act to Prevent Pollution from Ships,'' as recently amended by
Congress. These rules will implement Annex V of the International Convention
for the Prevention of Pollution by Ships, 1973 entitled ``Regulations for
the Prevention of Pollution by Garbage by Ships.'' These regulations would
apply to marine craft of any size or type, including commercially operated
ships and fishing vessels, uninspected vessels, recreational boats, oil
rigs and platforms. These proposed rules also specify that ports and terminals,
including recreational boat

[remaining text deleted...]
09:17       USER: "Yes. Anything to get sommething going here."
09:18       USER: Mark as Relevant document ranked 3 with document ID FR881027-0027
09:20       USER: Display Full Text of Document ranked 4

09:21       SYS:
                        Part III: The Catastrophe A Great Sea Is Dying; `Hundreds Will Die With It' LaserPhoto NY307 of July 24; Graphics SOVNEWSO3, SOVNEWSOL
AP900724-0203
BUKHARA, U.S.S.R. (AP)

   A tragedy vast in scope is unfolding on the arid
plains of Central Asia, a region already unsettled by stirrings of
dissent and nationalism. This is the third article in a four-part
series based on an AP special correspondent's journey through the
Soviet south.
 By MORT ROSENBLUM AP Special Correspondent
   Beyond Bukhara's irrigated deserts, the
noble Amu Darya river dwindles down to a pathetic, poisonous
trickle, a victim in one of the world's worst ecological disasters.
   Deprived of the Amu Darya and its sister stream, the Syr Darya,
the Aral Sea is dying. Within 30 years, what was the world's fourth
largest inland sea is expected to be only a dry bed of deadly salt.
   Scientists say the sea's condition reflects a far greater crisis
across much of Central Asia, caused by years of Soviet insistence
on

[remaining text deleted...]

09:49       USER: Scroll Full Text of Document to next matched keyword (10 times)

09:57       USER: "I don't like it. It's all about sea."
09:59       USER: Display Full Text of Document ranked 5

10:00       SYS:
                          SOVIET LAKE'S DEATH MIRRORED IN CALIFORNIA
SJMN91-06324011
  California News     Tuesday, November 19, 1991 00324011,SJ1
  Photo, Map; PHOTO: Aral Sea Information Committee; VISIT TO NEVADA -- Bill
Davoren, left, and Yusup Kamalov visit Pyramid Lake in Nevada. Kamalov is
visiting the United States to study ways to revive his homeland, the Aral Sea
region of the Soviet Union.; MAP: Mercury News; (Soviet Union, showing Aral
Sea)      For a group of Bay Area activists, the death of the Aral Sea in Soviet Central
Asia has unrolled like a nightmarish, speeded-up version of California
history.;    "What we took 150 years to destroy, they took only 30 years to
do," said Bill Davoren, Tiburon environmentalist and recent co-founder of a
U.S.-Soviet group dedicated to restoring what was once the fourth-largest
inland sea in the world.          Massive cotton irrigation projects have completely siphoned off river
flows, shrinking a sea famous as a teeming fishing center to a quarter of its

[remaining text deleted...]

10:06       USER: Scroll Full Text of Document to next matched keyword (12 times)

10:15       USER: "This is all about sea, sea, sea. No good."
10:17       USER: Display Full Text of Document ranked 6

10:18       SYS:
                        Safety Questions Over North Sea Oil Rig
AP880707-0139
ABERDEEN, Scotland (AP)
By BEN DOBBIN Associated Press Writer
```

```
   Roy Carey, a survivor of the world's
worst offshore oil platform disaster, said Thursday: ``We had all
that water around us and no way to get it to us. We had sprinklers
but they weren't strong enough.''
   Carey's remark as he lay badly burned in a hospital bed in
Aberdeen Royal Infirmary pointed up a bitter irony of offshore
oil-drilling, one of the world's most dangerous occupations.
   The presumed death toll from the explosion and fire Wednesday
night on the Piper Alpha platform in the North Sea exceeded 160.
   Urgent questions about safety in the offshore oil and gas
industry are being raised by the disaster, which brings to more
than 300 the number of people who have died in the British sector
of the North Sea since gas was first pumped ashore in 1967.
   Drilling accidents killed 76 and a helicopter crash killed 45.
Last year, six workers were killed an

[remaining text deleted...]
10:34       USER: "I'm using the keyword button here."
10:35       USER: Scroll Full Text of Document to next matched keyword (6 times)

10:39       USER: "Very frustrating."
10:41       USER: Display Full Text of Document ranked 7

10:45       SYS:

FR881024-0194

Federal Register  / Vol. 53, No. 205 / Monday, October 24, 1988
/ Unified Agenda  Federal Register  / Vol. 53, No. 205 / Monday, October 24, 1988
/ Unified Agenda  SUBJECT INDEX TO THE UNIFIED AGENDA OF FEDERAL REGULATIONS  Sequence Number Sequence Number Sequence Number Sequence Number  A
Accountants, peer review .....3982
 Accrual basis .....780
 Airline uniform system .....2068
 Automated clearing house .....2116
 Bills of lading .....3088
 Bonds .....2118, 2120
 Colleges and universities .....607, 3169
 Cost principles .....3166
 Emission standards .....2957
 Job training audit .....1424
 Loans .....3727
 Motor carriers .....3823
 Rail costing system .....3814
 Reclamation actions .....2117
 State and local governments .....417, 454, 3166
 Treasury checks .....2117
 Airline tickets .....1561
 American viticultural areas .....2132
 Athletes and athletic events .....2124
 Distilled spirits .....2123, 2124, 2140
 Food store practices .....3805

[remaining text deleted...]

10:58       USER: Scroll Full Text of Document to next matched keyword (3 times)

11:00       USER: "Deep sea bed mining program. That's interesting."
11:00       USER: Scroll Full Text of Document to next matched keyword (36 times)

11:29       USER: "Hm, bunches of fish. If I put this in I might get things with oceans and otters and scallops. Actually, it might be kind of interesting. I'm gonna try it."
11:28       USER: Scroll Full Text of Document to next matched keyword
11:31       USER: Mark as Relevant document ranked 7 with document ID FR881024-0194
11:32       USER: Display Full Text of Document ranked 8

11:33       SYS:
                         Oil Men Tend Garden On North Sea Oil Rig
AP880405-0032
LONDON (AP)

   Britons are such avid gardeners they even find
time for the hobby on an oil rig in the storm-tossed North Sea.
   Workers on the U.S.-owned Chevron Petroleum platform are growing
melons, oranges, herbs, vegetables and decorative pot plants in a
greenhouse, 100 miles off the Shetland Islands northeast of
Scotland.
   The 10-foot by 8-foot glass and aluminium greenhouse is bolted
to the deck and heated by electricity from the rig's
turbine-generators which run on natural gas pumped up from the
seabed.
   Pests which bother land-based gardeners are unknown at sea in
the northern latitude, reported London's Daily Mail newspaper
Tuesday, beside a picture of the little greenhouse above the waves
in the Ninian oilfield.
   The greenhouse cost $750, half from the rig's 18-member garden
club and half from the company.
[remaining text deleted...]

11:35       USER: "Gardeners, no."
11:36       USER: Display Full Text of Document ranked 9

11:36       SYS:
                         Three More Bodies Found In North Sea Platform Disaster With PM-Oil City, Bjt
AP880711-0049
ABERDEEN, Scotland (AP)

   The Piper Alpha oil and gas platform
was still burning and smoking early today, five days after
explosions and fire on the North Sea rig killed 166 men in the
world's worst oilfield disaster.
   The semi-submersible support rig, Tharos, found three more
bodies by early today, bringing the total recovered to 20.
   Texas oilfield fire and blowout specialist Red Adair, 73, and
his team waited on the rig to board the tilting, ruined platform.
Adair's team said they had planned to return to the platform today
to continue clearing debris before trying to cap a blazing wellhead.
   But a spokeswoman for the platform's operators, Occidental
Petroleum Corp., said Adair's plans were put on hold by 35 mph
winds and forecasts of possible lightning and thunderstorms.
```

Adair said three or four of the platform's 36 wellheads were still burning, two of them sending up flames

[remaining text deleted...]

11:40 USER: "That's not gonna help me."
11:41 USER: Display Full Text of Document ranked 10

11:43 SYS:
Automated all-weather cargo transfer system
PT3-05154561
Lee; Donald E. 2033 Parkview Ave. Abington PA 19001
An advanced automated vessel cargo transfer system for loading and unloading of ships and lighters. It includes an articulated manipulator arm mounted on a frame. The arm is provided with a spreader bar at the distal end thereof. The spreader bar is provided with facilities for grasping cargo. Sensors track the movement of the vessel, and automatically responsive controllers adjust the motion and position of the spreader bar to follow the motion of the vessel. Berthing modules are provided to aid in controlling the motion of the vessel. In a major embodiment, the manipulator arm is mounted on a transverse frame that bridges spaced-apart floating barges, and provisions are made for serving vessels both on the outboard and inboard sides of the barges. In another embodiment, the manipulator a

[remaining text deleted...]

11:48 USER: Scroll Full Text of Document to next matched keyword

11:48 USER: "I think I'm gonna have to put ocean here."
11:48 USER: Scroll Full Text of Document to next matched keyword (19 times)

12:00 USER: "That was number 10. I'm gonna rerun this query and then I'm gonna try ocean."
12:05 USER: Run Query

12:05 USER: "My time is still fine."
12:05 SYS: Submitted Query:
laws of the sea
and 4 marked documents

12:15 USER: "It's slow. Either it's a strange query or something's happening. Hm. Ah, there we go."
13:10 SYS: Result of Query Evaluation: 150 documents retrieved
13:14 SYS: Display of Titles of Retrieved Documents:

```
 R NS 1.  FR881027-0027
 R NS 2.  AP881101-0176    U.S. Votes Against Law Of Sea Convention
NR NS 3.  AP900201-0221    Dead Sea Was an Active Trade Route as Far Back as 2,300 Years
NR NS 4.  AP900724-0203    Part III: The Catastrophe A Great Sea Is Dying; `Hundreds Will Die With It' LaserPhoto NY307 of July 24; Graphics
SOVNEWSO3, SOVNEWSOL
NR NS 5.  SJMN91-06324011    SOVIET LAKE'S DEATH MIRRORED IN CALIFORNIA
 R NS 6.  FR881024-0194
NR NS 7.  AP901113-0077    Monterey Looks for Federal Law to Help Evict Sea Lions
NR NS 8.  AP881031-0251   AGENCIES AND RADIO OUT From AP Newsfeatures
NR NS 9.  FR88105-0063
NR NS 10.  AP900901-0010    Death Threats Over Marine Mammals
```

13:37 USER: "Ok, this is not working. I will look at 7,8,9 and 10, and then I'm putting oceans in here and I'll try again."

13:39 USER: Display Full Text of Document ranked 7

13:41 SYS:
Monterey Looks for Federal Law to Help Evict Sea Lions
AP901113-0077
MONTEREY, Calif. (AP)

The Monterey harbor has become a sort of Sea Lion Motel where the sardines are plenty and the living is easy. But the pinnipeds may soon face eviction as officials renew their fight to roust the mammals.

City officials hope to drive away the mammals by enforcing a proposed amendment to a 1972 federal law that protects sea lions from harm. The change, which could go into effect early next year, would make it illegal to feed marine mammals, putting an end to concessionaires selling sea lion snacks that tourists have used to lure the animals to shore.

``To the average Joe, the sea lions look like something out of a Walt Disney movie, but until you go down to the docks and stand right next to one of these monsters, it's hard to understand what the city is dealing with,'' said Mitch Whippern, administrative assistant for the Monterey Public Facilities Department.

Most of the hundred

[remaining text deleted...]

13:49 USER: Scroll Full Text of Document to next matched keyword (4 times)

13:53 USER: "That looks interesting."
13:55 USER: Scroll Full Text of Document to next matched keyword (9 times)

14:02 USER: "The system should help me here. I should be able to put in the documents I find relevant and it should be easy."
14:04 USER: Mark as Relevant document ranked 7 with document ID AP901113-0077
14:06 USER: Display Full Text of Document ranked 8

14:07 SYS:
AGENCIES AND RADIO OUT From AP Newsfeatures
AP881031-0251
SEATTLE (AP)

EDITOR'S NOTE _ Growing herds of California sea lions that converge on Puget Sound each fall are becoming a nuisance, raiding the fisheries and getting in the way of divers. It's a Catch 22 for environmentalists that the law intended to protect endangered marine mammals also has produced a threat to some endangered fish.
By KATIA BLACKBURN Associated Press Writer

Fishermen, divers and coastal residents on Puget Sound are being frustrated by large herds of sea lions and harbor seals feasting on bottom fish, salmon and endangered steelhead

```
trout headed upstream to spawn.
   The federal law that protects endangered whales and other marine
mammals prohibits killing the sea lions which kill and eat more
than half of the steelhead run each year.
   Up to 2,000 sea lions converge on Puget Sound each fall and
their numbers are on the rise, wildlife experts say.
   In addition to their threat to the fisher

[remaining text deleted...]

14:13       USER: "I am not doing good work on this."
14:15       USER: Mark as Relevant document ranked 8 with document ID AP881031-0251
14:19       USER: Display Full Text of Document ranked 9

14:19       SYS:

FR88105-0063

[Docket No. 87-29] Hemisphere Navigation Co., Inc. v. Sea-Land Service, Inc.; Filing
of Complaint and Assignment

Notice is given that a complaint filed by Hemisphere Navigation Co., Inc.
(``Hemisphere") against Sea-Land Service, Inc. (``Sea-Land") was served
December 29, 1987. Hemisphere alleges that Sea-Land has violated various
sections of the Intercoastal Shipping Act of 1933, 46 U.S.C. app. 843,
the Shipping Act, 1916, 46 U.S.C. app. 801, and the Shipping Act of 1984,
46 U.S.C. app. 1701, by wrongfully charging and demanding, collecting and
receiving ocean freight and terminal charges in excess of what was properly
applicable pursuant to Sea-Land's Tariff FMC-F-No. 61, in effect between
San Juan, Puerto Rico and U.S. Atlantic and Gulf ports.

This proceeding has been assigned to Administrative Law Judge Charles E.
Morgan (``Presiding Officer"). Hearing in this matter, if any is held,
shall commence within the time limitations prescribed in 46 CFR 502.61.
Th
[remaining text deleted...]

14:32       USER: "Here's number 9. I've marked 7 and 8 relevant."
14:33       USER: Mark as Relevant document ranked 9 with document ID FR88105-0063
14:35       USER:         Display Full Text of Document ranked 10

14:36       SYS:
                          Death Threats Over Marine Mammals
AP900901-0010
BANDON, Ore. (AP)
By JEFF BARNARD Associated Press Writer
   The idea of a floating hospital for injured
seals and sea lions seemed like a good one to Dan Deuel and Mary
Crumley. But it smacked of harboring the enemy to some people, who
have made death threats against the couple.
   ``Man is pretty possessive," said Neal Langbehn, a National
Marine Fisheries Service special agent, whose job includes tracking
down people who shoot marine mammals. ``If a critter causes him any
problems, all his natural instinct seems to be to get rid of the
critter. As man takes up more and more of the Earth, there are
going to be more conflicts."
   Deuel and Crumley and some volunteers run Free Flight Bird and
Marine Mammal Rehabilitation Ltd. from their home on a bluff
overlooking the Pacific Ocean.
   Earlier this year, Deuel suggested to the Bandon Port Commission
that he could move his operation to a barge in the harbor. That
would make it mo

[remaining text deleted...]

14:40       USER: Scroll Full Text of Document to next matched keyword ( 2 times)
14:44       USER: Mark as Relevant document ranked 10 with document ID AP900901-0010

14:45       USER: "I'll run this. But I'm gonna get something about federal laws about sea lions."
14:46       USER:         Run Query
14:46       SYS:          Submitted Query:
                          laws of the sea
                          and 8 marked documents

15:15       USER: "It's running my query. Taking its time. Very annoying."
15:36       SYS: Result of Query Evaluation: 150 documents retrieved

15:37       USER: "Well, it added some terms, it added 17 terms. I have a bad feeling I'm gonna get a lot of stuff about marine animals and sea mammals."
15:40       SYS: Display of Titles of Retrieved Documents:

                            R NS 1.  FR881027-0027
                            R NS 2.  AP881031-0251    AGENCIES AND RADIO OUT From AP Newsfeatures
                            R NS 3.  FR881024-0194
                            R NS 4.  AP901113-0077    Monterey Looks for Federal Law to Help Evict Sea Lions
                            R NS 5.  AP900901-0010    Death Threats Over Marine Mammals
                           NR NS 6.  AP880730-0024    No Syringes, But Western Swimmers Do Contend with Sharks, Sewage Spills With AM-Sea Scare, Bjt
                           NR NS 7.  SJMN91-06011130    DEEP-SEA TEST OF WASTE DUMPING PROPOSED SCIENTISTS SAY OCEANS COULD SPARE LAND, AIR
                           NR NS 8.  AP881103-0224    Italy Making a Start on Saving Seas Around It
                           NR NS 9.  AP901112-0005    Cruise Lines Trying to be Eco-Friendly with Garbage
                           NR NS 10.  AP880801-0020    Gumshoes On Garbage Zero In On Beach-Defiling Wastes LaserPhoto NY5

16:00       USER:" Actually, this is starting to look interesting. I'm adding ocean in here and running the query again to see what that does."
16:10       USER: Run Query
16:10       SYS:          Submitted Query:
                          laws of the sea ocean
                          and 8 marked documents
16:20       USER: Rereads topic description and instructions.

16:21       USER: "You can't talk to anyone, you have no idea of what they mean, it's hard for me being in a situation like this where you can't talk to a person and say is this
relevant    to you. A person who was interested in this question could use this database better than me."
17:00       SYS: Result of Query Evaluation: 150 documents retrieved
17:04       SYS: Display of Titles of Retrieved Documents:

                            R NS 1.  FR881027-0027
                            R NS 2.  AP881031-0251    AGENCIES AND RADIO OUT From AP Newsfeatures
                            R NS 3.  FR881024-0194
                            R NS 4.  AP900901-0010    Death Threats Over Marine Mammals
                           NR NS 5.  SJMN91-06011130    DEEP-SEA TEST OF WASTE DUMPING PROPOSED SCIENTISTS SAY OCEANS COULD SPARE LAND, AIR
```

```
                    R NS 6.  AP901113-0077   Monterey Looks for Federal Law to Help Evict Sea Lions
                    NR NS 7.  AP880730-0024   No Syringes, But Western Swimmers Do Contend with Sharks, Sewage Spills With AM-Sea Scare, Bjt
                    NR NS 8.  AP901112-0005   Cruise Lines Trying to be Eco-Friendly with Garbage
                    NR NS 9.  AP880801-0020   Gumshoes On Garbage Zero In On Beach-Defiling Wastes LaserPhoto NY5
                    NR NS 10.  FR88616-0064

17:20     USER: "My top five documents are still in the top six."
17:21     USER: Reads titles. "This is too environmental. Let me look at what I got."
17:22     USER: Display Full Text of Document ranked 1

17:30     SYS:

FR881027-0027

Federal Register   / Vol. 53, No. 208 / Thursday, October 27, 1988
/ Proposed Rules
 DEPARTMENT OF TRANSPORTATION Coast Guard 33 CFR Parts 151, 155 and 158 46 CFR Part 25  [CGD 88-002] RIN 2115-AC89 Regulations Implementing the Pollution
Prevention Requirements of
Annex V of MARPOL 73/78 AGENCY:  Coast Guard, DOT.
 ACTION:  Notice of proposed rulemaking.
 SUMMARY:  The Coast Guard proposes rules to implement the requirements
of the ``Act to Prevent Pollution from Ships," as recently amended by
Congress. These rules will implement Annex V of the International Convention
for the Prevention of Pollution by Ships, 1973 entitled ``Regulations for
the Prevention of Pollution by Garbage by Ships." These regulations would
apply to marine craft of any size or type, including commercially operated
ships and fishing vessels, uninspected vessels, recreational boats, oil
rigs and platforms. These proposed rules also specify that ports and terminals,
including recreational boat

[remaining text deleted...]

17:53     USER: "I could try regulation, too."
17:54     USER: Scroll List of Titles of Retrieved Documents
17:58     SYS:  Display of Titles of Retrieved Documents:

                    NR NS 11.  FR88518-0026
                    NR NS 12.  FR88506-0038
                    NR NS 13.  FR88927-0030
                    NR NS 14.  AP881103-0224   Italy Making a Start on Saving Seas Around It
                    NR NS 15.  SJMN91-06015044    WILSON'S BACKING ON REFUGE SOUGHT
                    NR NS 16.  AP900203-0031    Washington State Wants to Kill Troublesome Sea Lions
                    NR NS 17.  AP880824-0210    Seal Deaths An Early Warning For Humans, Scientists Say
                    NR NS 18.  WSJ910213-0077       A Fragile Ecosystem    Off Bahrain Faces    Ravages of Oil Slick   ---   It Abounds With Marine Life    That
Fishermen           Rely On;    The Shy Dugong's Peril   ----   By Ken Wells    Staff Reporter of The Wall Street Journal
                    NR NS 19.  AP880518-0300    Illegal Foreign Fishing in Bering Sea Strains Coast Guard Patrols An AP Extra
                    NR NS 20.  AP880511-0024    Fisheries, Environmentalists Agree On Truce Over Coastal Fishing

17:58     USER: Scroll List of Titles of Retrieved Documents
18:01     SYS: Display of Titles of Retrieved Documents:

                    NR NS 21.  AP880629-0098    In Great lakes, Too, Plastic Junk Claims Victims
                    NR NS 22.  AP880623-0239    In Great lakes, Too, Plastic Junk Claims Victims
                    NR NS 23.  AP900616-0073    Texas Beaches Are Important for Wildlife
                    NR NS 24.  SJMN91-06205161     6 DRUG SUSPECTS BURNED SHIP, JUMPED IN SEA
                    NR NS 25.  FR88504-0025
                    NR NS 26.  FR88121-0023
                    NR NS 27.  AP900319-0109    Sewage Dumped at Sea Poses Health Risks to Swimmers, Seafood Lovers
                    NR NS 28.  SJMN91-06193189     DEVICE MAY BLOW WHISTLE ON MONTEREY'S SEA LIONS
                    NR NS 29.  AP880725-0215   AGENCIES AND RADIO OUT From AP Newsfeatures
                    NR NS 30.  AP880812-0015    Congress Moves On Medical Waste Issue Before Returning Home For Recess

17:45     USER: " Garbage. I lost my whole law of the sea train. This is all about garbage. No good. Garbage and sea lions. I could fail at this. This is really bad."
18:10     USER: Scroll List of Titles of Retrieved Documents
18:13     SYS:  Display of Titles of Retrieved Documents:

                    R NS 1.  FR881027-0027
                    R NS 2.  AP881031-0251    AGENCIES AND RADIO OUT From AP Newsfeatures
                    R NS 3.  FR881024-0194
                    R NS 4.  AP900901-0010    Death Threats Over Marine Mammals
                    NR NS 5.  SJMN91-06011130    DEEP-SEA TEST OF WASTE DUMPING PROPOSED SCIENTISTS SAY OCEANS COULD SPARE LAND, AIR
                    R NS 6.  AP901113-0077   Monterey Looks for Federal Law to Help Evict Sea Lions
                    NR NS 7.  AP880730-0024   No Syringes, But Western Swimmers Do Contend with Sharks, Sewage Spills With AM-Sea Scare, Bjt
                    NR NS 8.  AP901112-0005   Cruise Lines Trying to be Eco-Friendly with Garbage
                    NR NS 9.  AP880801-0020   Gumshoes On Garbage Zero In On Beach-Defiling Wastes LaserPhoto NY5
                    NR NS 10.  FR88616-0064

18:16     USER: Undo Relevance Marking for document ranked 1 with document ID FR881027-0027
18:16     USER: "That's out of there."
18:18     USER: Display Full Text of Document ranked 2

18:19     SYS:
                    AGENCIES AND RADIO OUT From AP Newsfeatures
AP881031-0251
SEATTLE (AP)


  EDITOR'S NOTE _ Growing herds of California sea lions that
converge on Puget Sound each fall are becoming a nuisance, raiding
the fisheries and getting in the way of divers. It's a Catch 22 for
environmentalists that the law intended to protect endangered
marine mammals also has produced a threat to some endangered fish.
 By KATIA BLACKBURN Associated Press Writer
   Fishermen, divers and coastal residents on Puget
Sound are being frustrated by large herds of sea lions and harbor
seals feasting on bottom fish, salmon and endangered steelhead
trout headed upstream to spawn.
   The federal law that protects endangered whales and other marine
mammals prohibits killing the sea lions which kill and eat more
than half of the steelhead run each year.
   Up to 2,000 sea lions converge on Puget Sound each fall and
their numbers are on the rise, wildlife experts say.
   In addition to their threat to the fisher

[remaining text deleted...]

18:27     USER: "Fisheries. Sea lions. No, sea lions have to go. This is really bad. I'm getting rid of the sea lion stuff."
18:30     USER: Undo Relevance Marking for document ranked 2 with document ID AP881031-0251
```

```
18:32        USER: Display Full Text of Document ranked 3

18:38        SYS:

FR881024-0194

Federal Register  / Vol. 53, No. 205 / Monday, October 24, 1988
/ Unified Agenda  Federal Register  / Vol. 53, No. 205 / Monday, October 24, 1988
/ Unified Agenda  SUBJECT INDEX TO THE UNIFIED AGENDA OF FEDERAL REGULATIONS  Sequence Number Sequence Number Sequence Number Sequence Number  A
Accountants, peer review .....3982
 Accrual basis .....780
 Airline uniform system .....2068
 Automated clearing house .....2116
 Bills of lading .....3088
 Bonds .....2118, 2120
 Colleges and universities .....607, 3169
 Cost principles .....3166
 Emission standards .....2957
 Job training audit .....1424
 Loans .....3727
 Motor carriers .....3823
 Rail costing system .....3814
 Reclamation actions .....2117
 State and local governments .....417, 454, 3166
 Treasury checks .....2117
 Airline tickets .....1561
 American viticultural areas .....2132
 Athletes and athletic events .....2124
 Distilled spirits .....2123, 2124, 2140
 Food store practices .....3805

[remaining text deleted...]

18:45        USER: "Keep that Federal Register thing in there, but out go the marine mammals."
18:47        USER: Undo Relevance Marking for document ranked 4 with document ID AP900901-0010
18:47        USER: "Out go the marine mammals."
18:51        USER: Undo Relevance Marking for document ranked 6 with document ID AP901113-0077
18:51        USER: "Out go the sea lions."
18:52        USER: Mark as Relevant document ranked 6 with document ID AP901113-0077
18:52        USER: Undo Relevance Marking for document ranked 6 with document ID AP901113-0077

18:53        USER: "Somewhere in there is my law of the sea piece. I'm adding in regulation as a synonym for law. Laws, regulation, treaties. I'm searching just as I would in
Boolean.                       Laws of the sea, oceans, regulations, treaties, pacts, agreements, rivers, water. What the hell, go do it."
19:48        USER: Run Query
19:49        SYS:         Submitted Query:
                          laws of the sea oceans regulations treaties pacts agreements rivers water
                          and 4 marked documents
20:00        USER: "I probably could have just put in that law of the sea, too. But so what, it wouldn't have turned up anything."
20:10        USER: "I'm starting to run out of time here. Another few minutes and that's it. This is slow. It seems slow."

20:30        SYS:         Result of Query Evaluation: 150 documents retrieved
20:34        SYS:         Display of Titles of Retrieved Documents:

                           R NS 1.  FR881024-0194
                          NR NS 2.  FR881027-0027
                          NR NS 3.  FR881028-0026
                          NR NS 4.  FR88504-0025
                          NR NS 5.  FR88818-0051
                          NR NS 6.  FR88616-0064
                          NR NS 7.  FR88518-0026
                          NR NS 8.  FR88929-0045
                          NR NS 9.  FR88928-0020
                          NR NS 10.  SJMN91-06106167     NUCLEAR PLANT BALKS AT ACTING TO SAVE FISH SAN ONOFRE REGULATORS DEBATE DAMAGE STUDY

20:45        USER: "Well, my number one document is relevant. It's that Federal Register thing. I wonder, though, where my other relevant marked things are."
20:47        USER: Scroll List of Titles of Retrieved Documents
20:49        SYS:         Display of Titles of Retrieved Documents:

                          NR NS 11.  FR88219-0014
                          NR NS 12.  AP901114-0042     UNDATED: and 1992.
                          NR NS 13.  FR88129-0069
                          NR NS 14.  FR88728-0077
                          NR NS 15.  FR88211-0134
                          NR NS 16.  FR881110-0025
                          NR NS 17.  FR88824-0017
                          NR NS 18.  FR881025-0034
                          NR NS 19.  FR88826-0028
                          NR NS 20.  FR881109-0010

20:52        USER: "Interestingly enough, all of this stuff is from the Federal Register. It's a trick question, I guess."
20:56        USER: Scroll List of Titles of Retrieved Documents
20:58        SYS:         Display of Titles of Retrieved Documents:

                          NR NS 21.  FR881109-0021
                          NR NS 22.  FR88928-0134
                          NR NS 23.  FR88927-0030
                          NR NS 24.  FR88728-0117
                          NR NS 25.  FR881024-0087
                          NR NS 26.  AP901112-0005     Cruise Lines Trying to be Eco-Friendly with Garbage
                          NR NS 27.  FR881129-0116
                          NR NS 28.  FR88516-0016
                          NR NS 29.  FR88817-0151
                          NR NS 30.  FR88920-0015

21:00        USER: "There's that garbage cruise from the AP."
21:03        USER: Scroll List of Titles of Retrieved Documents
21:06        SYS:         Display of Titles of Retrieved Documents:

                          NR NS 31.  SJMN91-06178174     SPOILS FROM BOTTOM OF BAY COULD RESTORE SUISUN MARSH NOVEL DREDGING PLAN WOULD SPAWN
                          WETLAND
                          NR NS 32.  FR88121-0023
                          NR NS 33.  FR88603-0014
                          NR NS 34.  FR88818-0049
                          NR NS 35.  FR88831-0030
                          NR NS 36.  FR88927-0146
                          NR NS 37.  FR88818-0017
                          NR NS 38.  FR88721-0030
                          NR NS 39.  FR88715-0124
```

```
                    NR NS 40.  FR88923-0038

21:05       USER: "All Federal Register. I like it. It seems promising."
21:06       USER: Scroll List of Titles of Retrieved Documents
21:08       SYS:        Display of Titles of Retrieved Documents:

                    NR NS 41.  FR881107-0012
                    NR NS 42.  FR88216-0002
                    NR NS 43.  FR88211-0002
                    NR NS 44.  FR88927-0013
                    NR NS 45.  FR881116-0138
                    NR NS 46.  FR881101-0022
                    NR NS 47.  FR881118-0132
                    NR NS 48.  FR881020-0052
                    NR NS 49.  FR88707-0101
                    NR NS 50.  FR88609-0048

21:10       USER: Scroll List of Titles of Retrieved Documents
21:12       SYS:        Display of Titles of Retrieved Documents:

                    NR NS 51.  FR88701-0154
                    NR NS 52.  FR88106-0016
                    NR NS 53.  FR88504-0121
                    NR NS 54.  FR88708-0007
                    NR NS 55.  FR88513-0158
                    NR NS 56.  FR88906-0033
                    NR NS 57.  FR88816-0008
                    NR NS 58.  FR88816-0007
                    NR NS 59.  AP901029-0103    Conference Considers Ban on Toxic Ocean Dumping; US Wants More Study
                    NR NS 60.  SJMN91-06297043     NEARSHORE TRAWLING OUTLAWED HALIBUT REGULATION TAKES EFFECT IN 1993

21:15       USER: "I think what I'm gonna do with this is look at some of the top ones. I want to make sure that I get that law-of-sea."

21:31       USER: Scroll List of Titles of Retrieved Documents (3 times)
21:34       SYS:        Display of Titles of Retrieved Documents:

                    NR NS 8.  FR88929-0045
                    NR NS 9.  FR88928-0020
                    NR NS 10.  SJMN91-06106167     NUCLEAR PLANT BALKS AT ACTING TO SAVE FISH SAN ONOFRE REGULATORS DEBATE DAMAGE STUDY
                    NR NS 11.  FR88219-0014
                    NR NS 12.  AP901114-0042    UNDATED: and 1992.
                    NR NS 13.  FR88129-0069
                    NR NS 14.  FR88728-0077
                    NR NS 15.  FR88211-0134
                    NR NS 16.  FR881110-0025
                    NR NS 17.  FR88824-0017

21:36       USER: Scroll List of Titles of Retrieved Documents (4 times)
21:38       SYS:        Display of Titles of Retrieved Documents:

                     R NS 1.  FR881024-0194
                    NR NS 2.  FR881027-0027
                    NR NS 3.  FR881028-0026
                    NR NS 4.  FR88504-0025
                    NR NS 5.  FR88818-0051
                    NR NS 6.  FR88616-0064
                    NR NS 7.  FR88518-0026
                    NR NS 8.  FR88929-0045
                    NR NS 9.  FR88928-0020
                    NR NS 10.  SJMN91-06106167     NUCLEAR PLANT BALKS AT ACTING TO SAVE FISH SAN ONOFRE REGULATORS DEBATE DAMAGE STUDY
21:49       USER:  Adds query terms.
21:52       USER:  Run Query
21:52       SYS:        Submitted Query:
                    laws of the sea oceans regulations treaties pacts agreements rivers water law-of-sea
                    and 4 marked documents
22:20       USER: "This is the worst search I've ever done in my life. It feels totally unnatural. It seems sloppy. I could potentially get really great stufff this way. But what
would happen            if I had a million documents? Several million. Can you afford to be this sloppy? Can the technology keep pace with you? If I was on the network
would it be even slower?            And, why am I doing this? Who's asking this question, and why are they asking it? It's beyond my imagination."

22:37       SYS: Result of Query Evaluation: 150 documents retrieved
22:42       SYS: Display of Titles of Retrieved Documents:

                     R NS 1.  FR881024-0194
                    NR NS 2.  FR881027-0027
                    NR NS 3.  FR881028-0026
                    NR NS 4.  FR88504-0025
                    NR NS 5.  FR88818-0051
                    NR NS 6.  FR88616-0064
                    NR NS 7.  FR88518-0026
                    NR NS 8.  FR88929-0045
                    NR NS 9.  FR88928-0020
                    NR NS 10.  FR88219-0014

22:44       USER: "Well, the number one document is still the number one document."
22:46       USER: Save document ranked 5 with document ID FR88818-0051
22:48       USER: Undo Save for document ranked 5 with document ID FR88818-0051
22:49       USER: Scroll List of Titles of Retrieved Documents
22:52       SYS:        Display of Titles of Retrieved Documents:

                    NR NS 11.  AP901114-0042    UNDATED: and 1992.
                    NR NS 12.  SJMN91-06106167     NUCLEAR PLANT BALKS AT ACTING TO SAVE FISH SAN ONOFRE REGULATORS DEBATE DAMAGE STUDY
                    NR NS 13.  FR88129-0069
                    NR NS 14.  FR88728-0077
                    NR NS 15.  FR88211-0134
                    NR NS 16.  FR881110-0025
                    NR NS 17.  FR88824-0017
                    NR NS 18.  FR881025-0034
                    NR NS 19.  FR881109-0010
                    NR NS 20.  FR881109-0021

22:56       USER: "And the Federal Register is still the source of choice. There are a couple of newspaper articles in the top twenty. But that law of the sea type thing, I have no
idea where        it is."
22:58       USER: Scroll List of Titles of Retrieved Documents
23:00       SYS:        Display of Titles of Retrieved Documents:

                    NR NS 21.  FR88826-0028
                    NR NS 22.  FR88928-0134
                    NR NS 23.  FR88927-0030
                    NR NS 24.  AP901112-0005    Cruise Lines Trying to be Eco-Friendly with Garbage
                    NR NS 25.  FR88728-0117
                    NR NS 26.  FR88516-0016
                    NR NS 27.  FR88121-0023
                    NR NS 28.  FR881024-0087
                    NR NS 29.  FR88817-0151
                    NR NS 30.  FR881129-0116

23:01       USER: "Garbage cruise. Still, all Federal Register."
23:05       USER: Scroll List of Titles of Retrieved Documents
23:08       SYS:        Display of Titles of Retrieved Documents:

                    NR NS 31.  FR88920-0015
                    NR NS 32.  SJMN91-06178174     SPOILS FROM BOTTOM OF BAY COULD RESTORE SUISUN MARSH NOVEL DREDGING PLAN WOULD SPAWN
                    WETLAND
                    NR NS 33.  FR88603-0014
                    NR NS 34.  FR88831-0030
                    NR NS 35.  FR88927-0146
                    NR NS 36.  FR88818-0049
                    NR NS 37.  FR88721-0030
                    NR NS 38.  FR88715-0124
                    NR NS 39.  FR881107-0012
                    NR NS 40.  FR88923-0038

23:06       USER: "San Jose Mercury story. It's still only pulling up 150 documents."
23:11       USER: Scroll List of Titles of Retrieved Documents
23:14       SYS:  Display of Titles of Retrieved Documents:

                    NR NS 41.  FR881116-0138
                    NR NS 42.  FR88927-0013
                    NR NS 43.  FR88818-0017
                    NR NS 44.  FR88707-0101
                    NR NS 45.  FR881118-0132
                    NR NS 46.  FR881020-0052
                    NR NS 47.  FR88701-0154
                    NR NS 48.  FR881101-0022
                    NR NS 49.  FR88609-0048
                    NR NS 50.  FR88216-0002

23:17       USER: "Ok, I'm coming back to my number one article. I think I'm gonna scan about ten of them. I don't know what the relevance feedback is adding, and I wonder if
there's            a place to take a look at that. I should have paid more attention to the tutorial. Oh, well."

23:18       USER: Scroll List of Titles of Retrieved Documents
23:22       SYS:        Display of Titles of Retrieved Documents:

                     R NS 1.  FR881024-0194
                    NR NS 2.  FR881027-0027
                    NR NS 3.  FR881028-0026
                    NR NS 4.  FR88504-0025
                    NR NS 5.  FR88818-0051
                    NR NS 6.  FR88616-0064
                    NR NS 7.  FR88518-0026
                    NR NS 8.  FR88929-0045
                    NR NS 9.  FR88928-0020
                    NR NS 10.  FR88219-0014

23:30       USER: Display Full Text of Document ranked 1
23:35       USER: "It's not even bringing up the documents quick."
23:40       SYS:
FR881024-0194

Federal Register  / Vol. 53, No. 205 / Monday, October 24, 1988
/ Unified Agenda  Federal Register  / Vol. 53, No. 205 / Monday, October 24, 1988
/ Unified Agenda  SUBJECT INDEX TO THE UNIFIED AGENDA OF FEDERAL REGULATIONS  Sequence Number Sequence Number Sequence Number Sequence Number  A
Accountants, peer review .....3982
 Accrual basis .....780
 Airline uniform system .....2068
 Automated clearing house .....2116
 Bills of lading .....3088
 Bonds .....2118, 2120
 Colleges and universities .....607, 3169
 Cost principles .....3166
 Emission standards .....2957
 Job training audit .....1424
 Loans .....3727
 Motor carriers .....3823
 Rail costing system .....3814
 Reclamation actions .....2117
 State and local governments .....417, 454, 3166
 Treasury checks .....2117
 Airline tickets .....1561
 American viticultural areas .....2132
 Athletes and athletic events .....2124
 Distilled spirits .....2123, 2124, 2140
 Food store practices .....3805

[remaining text deleted...]
23:42       USER: "Here's this table of contents.Subject indexed to the unified agenda of federal regulations. I don't know what it is."


23:55       USER: Scroll Full Text of Document to next matched keyword (24 times)

24:12       USER: "I have no idea what I've done here."
24:15       USER: Display Full Text of Document ranked 2
24:19       SYS:

FR881027-0027
Federal Register   / Vol. 53, No. 208 / Thursday, October 27, 1988
/ Proposed Rules
 DEPARTMENT OF TRANSPORTATION Coast Guard 33 CFR Parts 151, 155 and 158 46 CFR Part 25  [CGD 88-002] RIN 2115-AC89 Regulations Implementing the Pollution
Prevention Requirements of
Annex V of MARPOL 73/78 AGENCY:  Coast Guard, DOT.
 ACTION:  Notice of proposed rulemaking.
 SUMMARY:  The Coast Guard proposes rules to implement the requirements
of the ``Act to Prevent Pollution from Ships," as recently amended by
Congress. These rules will implement Annex V of the International Convention
for the Prevention of Pollution by Ships, 1973 entitled ``Regulations for
the Prevention of Pollution by Garbage by Ships." These regulations would
```

```
apply to marine craft of any size or type, including commercially operated
ships and fishing vessels, uninspected vessels, recreational boats, oil
rigs and platforms. These proposed rules also specify that ports and terminals,
including recreational boat

[remaining text deleted...]
24:32         USER: "I gotta stop."
24:32         USER: Display Full Text of Document ranked 3
24:35         SYS:

FR881028-0026
Federal Register  / Vol. 53, No. 209 / Friday, October 28, 1988
/ Rules and Regulations
 DEPARTMENT OF COMMERCE  National Oceanic and Atmospheric Administration 15 CFR Part 922  [Docket No. 60469-7005]  National Marine Sanctuary Program Regulations
AGENCY:  Office of Ocean and Coastal Resource Management (OCRM),
National Ocean Service (NOS), National Oceanic and Atmospheric Administration
(NOAA), Commerce.
 ACTION:  Final rule.
 SUMMARY:  These final regulations implement the provisions of the
Marine Sanctuaries Amendments of 1984, Title I of Pub. L. 98-498 (16 U.S.C.
1431
 et seq. ) (the Act or the Amendments). While the Amendments build
upon the foundation established in the previous Marine Sanctuary Program
regulations (48 FR 24296 (1983)), revisions are necessary to reflect procedural
and some policy changes of the Amendments.
 DATE:  These regulations are effective November 28, 1988.
 FOR FURTHER INFORMATION CONTACT: Joseph A. Uravitch, Chief, Marine
and E

[remaining text deleted...]
24:44         USER: Display Full Text of Document ranked 4
24:46         SYS:

FR88504-0025
Federal Register  / Vol. 53, No. 86 / Wedneday, May 4, 1988 / Rules
and Regulations
 DEPARTMENT OF COMMERCE National Oceanic and Atmospheric Administration 50 CFR Part 661 [Docket No. 80482-8082] Ocean Salmon Fisheries off the Coasts of
Washington, Oregon, and California AGENCY:  National Marine Fisheries Service (NMFS), NOAA, Commerce.
 ACTION:  Emergency interim rule; request for comments.
 SUMMARY:  The Secretary of Commerce (Secretary) issues an emergency
interim rule to establish fishery management measures for the commercial
and recreational ocean salmon fisheries off Washington, Oregon, and California
for 1988. The management measures are intended to prevent overfishing and
to apportion the ocean harvest equitably among non-Indian commercial and
recreational and treaty Indian fisheries. The regulations also are calculated
to allow a portion of the salmon runs to escape the ocean fisheries to
provide for Indian and non-Indian inside fisheries and spawning. Most

[remaining text deleted...]
24:59         USER: "I suppose that if you looked through these you could find out what are the laws of the sea, and are they uniform. But, was I supposed to find news stories?"

25:00         USER: Display Full Text of Document ranked 5
25:06         SYS:

FR88818-0051
Federal Register  / Vol. 53, No. 160 / Thursday, August 18, 1988
/ Proposed Rules
 DEPARTMENT OF THE INTERIOR Minerals Management Service 30 CFR Part 282 Operations in the Outer Continental Shelf for Minerals Other Than
Oil, Gas, and Sulphur AGENCY:  Minerals Management Service, Interior.
 ACTION:  Proposed rule.
 SUMMARY:  The proposed rule would establish a separate set of general
regulations designed to govern postlease discovery, delineation, development,
and production of minerals other than oil, gas, and sulphur within the
Outer Continental Shelf (OCS) of the United States. The proposed rule recognizes
the special circumstances, issues, and requirements associated with those
OCS minerals. It establishes practices and procedures for wise management
of OCS resources, allowing balanced orderly postlease discovery, delineation,
development, and production of minerals other than oil, gas, and sulphur,
while protecting the human, marine, and coastal environments; pre

[remaining text deleted...]
25:25         USER: "I like it. It's good. But I would feel a lot more comfortable with news stories about it. No news stories are coming up."

25:25         USER: Display Full Text of Document ranked 6
25:26         SYS:

FR88616-0064
National Oceanic and Atmospheric Administration [Docket No. 80520-8120] Foreign Fishing; Bering Sea Fishermen's Association AGENCY:  National Marine Fisheries Service
(NMFS), NOAA, Commerce.
 ACTION:  Notice of decision on petition for rulemaking; Bering
Sea Fishermen's Association.
 SUMMARY:  NOAA publishes notice of its decision on a petition for
rulemaking submitted by the Bering Sea Fisherman's Association. The notice
summarizes the comments received and NOAA's decision not to undertake the
rulemaking requested by the petition at this time. The agency continues
to work to develop a comprehensive solution to address issues relating
to the harvest of U.S. exclusive economic zone (EEZ) resources from waters
beyond the EEZ.
 SUPPLEMENTARY INFORMATION:  Background

NOAA published a notice of receipt of a petition for rulemaking submitted
by the Bering Sea Fishermen's Association on November 5, 1987 (52 FR 42469).
The petition asked the United States Department of C

[remaining text deleted...]
25:40         USER: "I'm going to undo relevance feedback on that document. I better hurry up."
25:42         USER: Undo Relevance Marking for document ranked 1 with document ID FR881024-0194
26:05         USER: Run Query
26:05         SYS:          Submitted Query:
                            laws of the sea oceans regulations treaties pacts agreements rivers water law-of-sea
                            and 3 marked documents
26:15         SYS:          Result of Query Evaluation: 150 documents retrieved

26:15         USER: "It doesn't make any bit of difference."
26:17         SYS:          Display of Titles of Retrieved Documents:

                             R NS 1.  AP881101-0176    U.S. Votes Against Law Of Sea Convention
                             NR NS 2.  FR881027-0027
```

```
                         NR NS 3.  FR881024-0194
                         NR NS 4.  AP900724-0203    Part III: The Catastrophe A Great Sea Is Dying; `Hundreds Will Die With It' LaserPhoto NY307 of July 24; Graphics
                         SOVNEWSO3, SOVNEWSOL
                         NR NS 5.  FR88616-0064
                          R NS 6.  FR88105-0063
                         NR NS 7.  FR881028-0026
                         NR NS 8.  SJMN91-06324011     SOVIET LAKE'S DEATH MIRRORED IN CALIFORNIA
                         NR NS 9.  FR88518-0026
                         NR NS 10.  FR88818-0051

26:35       USER: "Huh. There's my law of sea convention article. And another relevant article pops up as well. I don't know what this other thing up here is. And I get more
newspaper       articles."
26:36       USER: Display Full Text of Document ranked 6
26:37       SYS:

FR88105-0063
[Docket No. 87-29] Hemisphere Navigation Co., Inc. v. Sea-Land Service, Inc.; Filing
of Complaint and Assignment

Notice is given that a complaint filed by Hemisphere Navigation Co., Inc.
(``Hemisphere") against Sea-Land Service, Inc. (``Sea-Land") was served
December 29, 1987. Hemisphere alleges that Sea-Land has violated various
sections of the Intercoastal Shipping Act of 1933, 46 U.S.C. app. 843,
the Shipping Act, 1916, 46 U.S.C. app. 801, and the Shipping Act of 1984,
46 U.S.C. app. 1701, by wrongfully charging and demanding, collecting and
receiving ocean freight and terminal charges in excess of what was properly
applicable pursuant to Sea-Land's Tariff FMC-F-No. 61, in effect between
San Juan, Puerto Rico and U.S. Atlantic and Gulf ports.

This proceeding has been assigned to Administrative Law Judge Charles E.
Morgan (``Presiding Officer"). Hearing in this matter, if any is held,
shall commence within the time limitations prescribed in 46 CFR 502.61.
Th

[remaining text deleted...]

26:41       USER: "I'm gonna get rid of number 6."
26:42       USER: Undo Relevance Marking for document ranked 6 with document ID FR88105-0063

26:42       USER: "I'll scan down and see if anything else is marked relevant, in the top 20, 30 or so."
26:44       USER: Scroll List of Titles of Retrieved Documents
26:47       SYS:        Display of Titles of Retrieved Documents:

                         NR NS 11.  AP900201-0221    Dead Sea Was an Active Trade Route as Far Back as 2,300 Years
                         NR NS 12.  FR88504-0025
                         NR NS 13.  AP901114-0042    UNDATED: and 1992.
                         NR NS 14.  AP881031-0251    AGENCIES AND RADIO OUT From AP Newsfeatures
                         NR NS 15.  PT3-05154561      Automated all-weather cargo transfer system
                         NR NS 16.  AP880816-0208    Nile Threatening To Flood Khartoum
                         NR NS 17.  FR88927-0030
                         NR NS 18.  AP900901-0010    Death Threats Over Marine Mammals
                         NR NS 19.  AP901220-0197    Echoes from the Deep: Clues to Changes in Sea Level
                         NR NS 20.  SJMN91-06185080     AQUARIUM DRIVES WONDERS OF THE SEA TO LABOR-CAMP KIDS

26:49       USER: Scroll List of Titles of Retrieved Documents
26:52       SYS:        Display of Titles of Retrieved Documents:

                         NR NS 21.  SJMN91-06226074     THE SEARCH FOR SEA LEVEL FOR RESEARCHERS OF GLOBAL WARMING, THE OCEAN'S HEIGHT PROVES
                         AN ELUSIVE MEASURE
                         NR NS 22.  SJMN91-06015044     WILSON'S BACKING ON REFUGE SOUGHT
                         NR NS 23.  AP880623-0017    After Nearly Five Months Adrift, Fishermen Say Sea Still Beckons
                         NR NS 24.  AP880817-0026    US Research Vessel Makes Breakthroughs in Restricted Soviet Zone
                         NR NS 25.  SJMN91-06178297     STANFORD GEOLOGY PROF CITED
                         NR NS 26.  AP900622-0055    Administrative Law Judge Fines Fishermen in Turtle Case
                         NR NS 27.  WSJ910213-0077          A Fragile Ecosystem     Off Bahrain Faces     Ravages of Oil Slick     ---     It Abounds With Marine Life     That
Fishermen                Rely On;    The Shy Dugong's Peril    ----    By Ken Wells     Staff Reporter of The Wall Street Journal
                         NR NS 28.  AP881110-0125    Scientists Discover New Undersea Hot Springs
                         NR NS 29.  AP900203-0031    Washington State Wants to Kill Troublesome Sea Lions
                         NR NS 30.  SJMN91-06242127     A WILD RIDE OFF THE CANADA COAST

26:53       USER: "I know that there's one other relevant document lurking there somewhere, but I don't have time to scan down the list. It would be nice if I didn't  have to scan
the list        to find it."
26:54       USER: Scroll List of Titles of Retrieved Documents (7 times)
26:58       SYS:        Display of Titles of Retrieved Documents:

                          R NS 1.  AP881101-0176    U.S. Votes Against Law Of Sea Convention
                         NR NS 2.  FR881027-0027
                         NR NS 3.  FR881024-0194
                         NR NS 4.  AP900724-0203    Part III: The Catastrophe A Great Sea Is Dying; `Hundreds Will Die With It' LaserPhoto NY307 of July 24; Graphics
                         SOVNEWSO3, SOVNEWSOL
                         NR NS 5.  FR88616-0064
                         NR NS 6.  FR88105-0063
                         NR NS 7.  FR881028-0026
                         NR NS 8.  SJMN91-06324011     SOVIET LAKE'S DEATH MIRRORED IN CALIFORNIA
                         NR NS 9.  FR88518-0026
                         NR NS 10.  FR88818-0051

27:01       USER: Run Query
27:02       SYS:        Submitted Query:
                        laws of the sea oceans regulations treaties pacts agreements rivers water law-of-sea
                        and 2 marked documents
27:09       SYS:        Result of Query Evaluation: 150 documents retrieved
27:11       SYS:        Display of Titles of Retrieved Documents:

                          R NS 1.  AP881101-0176    U.S. Votes Against Law Of Sea Convention
                         NR NS 2.  FR881027-0027
                         NR NS 3.  FR881024-0194
                         NR NS 4.  AP900724-0203    Part III: The Catastrophe A Great Sea Is Dying; `Hundreds Will Die With It' LaserPhoto NY307 of July 24; Graphics
                         SOVNEWSO3, SOVNEWSOL
                         NR NS 5.  FR881028-0026
                         NR NS 6.  FR88616-0064
                         NR NS 7.  FR88818-0051
                         NR NS 8.  SJMN91-06324011     SOVIET LAKE'S DEATH MIRRORED IN CALIFORNIA
                         NR NS 9.  FR88504-0025
                         NR NS 10.  AP900201-0221    Dead Sea Was an Active Trade Route as Far Back as 2,300 Years

27:25       USER: "I don't like it. There's lots of Federal Register stuff. Maybe there's a relationship between these documents."
27:26       USER: Scroll List of Titles of Retrieved Documents
```

```
27:28   SYS:    Display of Titles of Retrieved Documents:

                NR NS 11.  FR88518-0026
                NR NS 12.  AP901114-0042   UNDATED: and 1992.
                NR NS 13.  PT3-05154561    Automated all-weather cargo transfer system
                NR NS 14.  AP880816-0208   Nile Threatening To Flood Khartoum
                NR NS 15.  FR88927-0030
                NR NS 16.  AP881031-0251   AGENCIES AND RADIO OUT From AP Newsfeatures
                NR NS 17.  FR88817-0020
                NR NS 18.  AP880707-0139   Safety Questions Over North Sea Oil Rig
                NR NS 19.  WSJ910213-0077    A Fragile Ecosystem   Off Bahrain Faces   Ravages of Oil Slick   ---   It Abounds With Marine Life   That Fishermen Rely On;   The Shy Dugong's Peril   ----   By Ken Wells   Staff Reporter of The Wall Street Journal
                NR NS 20.  AP900901-0010   Death Threats Over Marine Mammals

27:40   USER: "From twelve down to twenty, this is no good. That Federal Register document is very helpful. It's the right combination of I don't know what."
27:47   USER: Scroll List of Titles of Retrieved Documents
27:50   SYS:    Display of Titles of Retrieved Documents:

                NR NS 21.  WSJ900514-0065    LAW   ----   By Milo Geyelin
                NR NS 22.  AP900622-0055   Administrative Law Judge Fines Fishermen in Turtle Case
                NR NS 23.  AP880711-0049   Three More Bodies Found In North Sea Platform Disaster With PM-Oil City, Bjt
                NR NS 24.  AP881029-0084   Venice Launches Important Stage Of Flood Control Plan
                NR NS 25.  AP901220-0197   Echoes from the Deep: Clues to Changes in Sea Level
                NR NS 26.  AP880817-0026   US Research Vessel Makes Breakthroughs in Restricted Soviet Zone
                NR NS 27.  AP880310-0236   Twenty-Three Nations Sign Treaty to Combat Terrorism at Sea
                NR NS 28.  FR88516-0048
                NR NS 29.  AP901029-0103   Conference Considers Ban on Toxic Ocean Dumping; US Wants More Study
                NR NS 30.  SJMN91-06105017    IT TAKES TWO TO TONGA HONORARY YACHTIES FALL IN WITH A BAD CROWD, AND EVERYTHING GOES DOWNHILL

27:51   USER: Scroll List of Titles of Retrieved Documents
27:53   SYS:    Display of Titles of Retrieved Documents:

                NR NS 31.  SJMN91-06222181    CAPTAIN'S BEHAVIOR WAS IRRESPONSIBLE
                NR NS 32.  AP900213-0069   Soviets Push for Expanded `Open Skies' Pact
                NR NS 33.  SJMN91-06185080    AQUARIUM DRIVES WONDERS OF THE SEA TO LABOR-CAMP KIDS
                NR NS 34.  FR88105-0063
                NR NS 35.  AP881025-0139   U.S.-Soviet Negotiators Call For Halt To Overfishing
                NR NS 36.  AP880708-0160   Reconstruction of Piper Alpha Disaster
                NR NS 37.  AP880729-0241   Scientists Find Light Source At Bottom Of Pacific
                NR NS 38.  SJMN91-06226074    THE SEARCH FOR SEA LEVEL FOR RESEARCHERS OF GLOBAL WARMING, THE OCEAN'S HEIGHT PROVES AN ELUSIVE MEASURE
                NR NS 39.  SJMN91-06228064    SEA WORLD PROBE HINGES ON AUTOPSY
                NR NS 40.  AP900504-0058   Driftnet Fishing Boats Spotted Bearing North Korean Flags

27:55   USER: Scroll List of Titles of Retrieved Documents
27:57   SYS:    Display of Titles of Retrieved Documents:

                NR NS 41.  FR88219-0014
                NR NS 42.  WSJ920323-0011    China Has Technology to Give Bombers   Long Flight Range, Worrying Neighbors   ----   By Nayan Chanda   Staff Reporter of The Wall Street Journal
                NR NS 43.  AP881110-0125   Scientists Discover New Undersea Hot Springs
                NR NS 44.  AP900203-0031   Washington State Wants to Kill Troublesome Sea Lions
                NR NS 45.  SJMN91-06117076    ANGER RUNS WHERE THE SALMON USED TO
                NR NS 46.  SJMN91-06015044    WILSON'S BACKING ON REFUGE SOUGHT
                NR NS 47.  AP880623-0017   After Nearly Five Months Adrift, Fishermen Say Sea Still Beckons
                NR NS 48.  SJMN91-06242127    A WILD RIDE OFF THE CANADA COAST
                NR NS 49.  WSJ920309-0003    Freer Markerts   Would Protect   Northwest Salmon   ----   By Zach Willey
                NR NS 50.  FR881109-0010

27:56   USER: "I feel lke an idiot, going down here looking for other potentially marked relevant documents. Oh, the screen tells how many there are. Only one."
27:58   USER: Scroll List of Titles of Retrieved Documents
28:01   SYS:    Display of Titles of Retrieved Documents:

                NR NS 51.  FR88121-0023
                NR NS 52.  AP900512-0030   Sea Lions Still Making Pier 39 Their Home LaserPhoto FX1
                NR NS 53.  AP880823-0088
                NR NS 54.  FR88506-0038
                NR NS 55.  FR88201-0084
                NR NS 56.  WSJ900822-0106    LEISURE &amp; ARTS -- Bookshelf:   What the Founding Fathers Really Wanted   ----   By Thomas P. Griesa
                NR NS 57.  AP880422-0066   Nile Delta Slowly Eroding Into Sea, Researcher Says
                NR NS 58.  SJMN91-06268085    BREAST-IMPLANT WARNING ULTRASOUND RECOMMENDED IN ADDITION TO MAMMOGRAM
                NR NS 59.  FR881024-0087
                NR NS 60.  AP881015-0132   Oil Rig Module Thought To Contain Bodies Raised To Surface

28:02   USER: Scroll List of Titles of Retrieved Documents
28:05   SYS:    Display of Titles of Retrieved Documents:

                NR NS 61.  AP880405-0032   Oil Men Tend Garden On North Sea Oil Rig
                NR NS 62.  AP900319-0109   Sewage Dumped at Sea Poses Health Risks to Swimmers, Seafood Lovers
                NR NS 63.  FR881202-0043
                NR NS 64.  AP880822-0251   Rising seas threaten pacific islands
                NR NS 65.  SJMN91-06178297    STANFORD GEOLOGY PROF CITED
                NR NS 66.  AP900928-0015   Water Well Suggests Ancient Man Wwas Smarter Than We Thought
                NR NS 67.  AP881012-0028   Plutonium Shipment Plans Questioned; Japan May Provide Armed Escort
                NR NS 68.  AP880922-0122   Oil Rig Fire Forces Evacuation; One Missing
                NR NS 69.  FR881206-0025
                NR NS 70.  AP900602-0072   Endangered Pelican and Sea Gull Eggs Stolen

28:05   USER: Scroll List of Titles of Retrieved Documents
28:07   SYS:    Display of Titles of Retrieved Documents:

                NR NS 71.  FR88928-0020
                NR NS 72.  PT3-05176471    Arrangement and method for protecting components in subsea systems
                NR NS 73.  WSJ900521-0089    U.S.-Soviet Summit to Yield Arms Pacts;   No Agreement in Sight on Europe, Trade   ----   By Walter S. Mossberg   Staff Reporter of The Wall Street Journal
                NR NS 74.  FR88831-0030
                NR NS 75.  AP900517-0140   Panel Calls For Excluder Devices On Trawlers To Save Endangered Sea Turtles
                NR NS 76.  AP880813-0154   Irrigation Killing Aral Sea, Report Says
                NR NS 77.  AP900129-0102   U.S. Ships Enter Soviet-Dominated Black Sea
                NR NS 78.  AP880927-0224   Foreign Submarines Still Elude Swedish Hunters
                NR NS 79.  FR88914-0104
                NR NS 80.  AP880319-0021   Adventurers Recount Unprecedented Row To Antarctica

28:07   USER: Scroll List of Titles of Retrieved Documents
28:10   SYS:    Display of Titles of Retrieved Documents:

                NR NS 81.  AP901113-0077   Monterey Looks for Federal Law to Help Evict Sea Lions
```

NR NS 82. WSJ910805-0086 Hong Kong Proposes a Sewage Strategy --- But Funding Uncertainty Threatens to Trash Plans ---- By Steve Glain Staff Reporter of The Wall Street Journal
NR NS 83. SJMN91-06193189 DEVICE MAY BLOW WHISTLE ON MONTEREY'S SEA LIONS
NR NS 84. SJMN91-06168108 TRADE THE HIKING BOOTS FOR SOME FLIPPERS MARINE PRESERVES PROTECT IMPORTANT COASTAL HABITATS
NR NS 85. FR881025-0034
NR NS 86. WSJ910723-0130 LEISURE &amp; ARTS: ...Humongous Marine Predators Underwater ---- By James P. Sterba
NR NS 87. AP880528-0058 With PM-Summit-Reagan Bjt Reagan, Gorbachev Expected To Sign Fisheries Pact
NR NS 88. FR88818-0049
NR NS 89. AP881116-0076 Scandinavian Ministers Adopt Plan to Rescue Seas
NR NS 90. AP880725-0215 AGENCIES AND RADIO OUT From AP Newsfeatures

28:10 USER: Scroll List of Titles of Retrieved Documents
28:13 SYS: Display of Titles of Retrieved Documents:

NR NS 91. WSJ900620-0136 Troubled Waters: Oil Tankers' Safety Is Assailed as Mishaps Average Four a Week - Ships Are Ill-Maintained, Some Crews Are Fatigued, Harbor Traffic Is a Mess - Game of `Chicken' in Houston By Caleb Solomon and Daniel Machalaba Staff Reporters of The Wall Street Journal
NR NS 92. FR88516-0016
NR NS 93. AP880725-0197 Venice Preparing to Test System to Protect It from Sea
NR NS 94. AP880315-0048 U.S.-Canada Free-Trade Pact To Help New England Economy, Study Says Handicapped People Protest DOT Transportation Policies NOAA Issues $100,000 In Fines For Illegal Fishing In Bering Sea Former Reagan Aide's Sentencing Postponed
NR NS 95. SJMN91-06161137 U.S. SENDS MEDICAL AID TO SOVIET REPUBLIC SUPPLIES FOR VICTIMS OF ARAL SEA POLLUTION
NR NS 96. FR88212-0146
NR NS 97. AP881224-0065 Loose Oil Storage Vessel Drifts 13 Miles
NR NS 98. AP880423-0166 U.S.-Soviet Talks On Bering Sea Adjourn With No Moratorium Agreement
NR NS 99. AP880526-0266 Mystery Oil Spills Endanger Ecology of Persian Gulf
NR NS 100. SJMN91-06226026 GLACIERS PRESENT A PUZZLE

28:14 USER: Scroll List of Titles of Retrieved Documents (7 times, back up to beginning of rank)
28:17 SYS: Display of Titles of Retrieved Documents:

R NS 1. AP881101-0176 U.S. Votes Against Law Of Sea Convention
NR NS 2. FR881027-0027
NR NS 3. FR881024-0194
NR NS 4. AP900724-0203 Part III: The Catastrophe A Great Sea Is Dying; `Hundreds Will Die With It' LaserPhoto NY307 of July 24; Graphics SOVNEWSO3, SOVNEWSOL
NR NS 5. FR881028-0026
NR NS 6. FR88616-0064
NR NS 7. FR88818-0051
NR NS 8. SJMN91-06324011 SOVIET LAKE'S DEATH MIRRORED IN CALIFORNIA
NR NS 9. FR88504-0025
NR NS 10. AP900201-0221 Dead Sea Was an Active Trade Route as Far Back as 2,300 Years

28:22 USER: "There's the Federal Register article. I'll mark it relevant and rerun my query."

28:24 USER: Mark as Relevant document ranked 3 with document ID FR881024-0194
28:26 USER: Run Query
28:26 SYS: Submitted Query:
laws of the sea oceans regulations treaties pacts agreements rivers water law-of-sea
and 3 marked documents

28:45 USER: "You put that Federal Register thing in there and it slows things down incredibly. How many terms does it add in? It just picks up a couple of extra terms. My query terms might have covered everything."
29:09 SYS: Result of Query Evaluation: 150 documents retrieved
29:13 SYS: Display of Titles of Retrieved Documents:

R NS 1. FR881024-0194
NR NS 2. FR881027-0027
NR NS 3. FR881028-0026
NR NS 4. FR88504-0025
NR NS 5. FR88818-0051
NR NS 6. FR88616-0064
NR NS 7. FR88518-0026
NR NS 8. FR88929-0045
NR NS 9. FR88928-0020
NR NS 10. SJMN91-06106167 NUCLEAR PLANT BALKS AT ACTING TO SAVE FISH SAN ONOFRE REGULATORS DEBATE DAMAGE STUDY
29:22 USER: "The law of the sea document disappears. Well, I don't care. This is going to have to be it."
29:23 USER: Scroll List of Titles of Retrieved Documents
29:26 SYS: Display of Titles of Retrieved Documents:

NR NS 11. FR88219-0014
NR NS 12. AP901114-0042 UNDATED: and 1992.
NR NS 13. FR88728-0077
NR NS 14. FR88129-0069
NR NS 15. FR88211-0134
NR NS 16. FR881110-0025
NR NS 17. FR88824-0017
NR NS 18. FR88826-0028
NR NS 19. FR881025-0034
NR NS 20. FR88928-0134

29:28 USER: Scroll List of Titles of Retrieved Documents
29:31 SYS: Display of Titles of Retrieved Documents:

NR NS 21. FR881109-0010
NR NS 22. FR881109-0021
R NS 23. AP881101-0176 U.S. Votes Against Law Of Sea Convention
NR NS 24. FR88728-0117
NR NS 25. FR88818-0017
NR NS 26. FR881024-0087
NR NS 27. FR881129-0116
NR NS 28. FR88818-0049
NR NS 29. AP901112-0005 Cruise Lines Trying to be Eco-Friendly with Garbage
NR NS 30. FR88216-0002

29:54 USER: "This is beyond my abilities. Ah, there it is, in the top 25. I think people could work with this. Something tells me the answer they would find is no, they're not uniform."

29:55 USER: Save Query
29:55 USER: "I'm not proud of it but I'm saving it."
30:01 USER: Query saved under name final

30:01 USER: Query read:
laws of the sea oceans regulations treaties pacts agreements rivers water law-of-sea

30:02 USER: with Relevance Feedback from 2 documents
30:08 USER: Stop Search

Is Recall Relevant? An Analysis of How User Interface Conditions affect Strategies and Performance in Large Scale Text Retrieval

Nipon Charoenkitkarn, Mark H. Chignell, and Gene Golovchinsky
Department of Industrial Engineering
University of Toronto

Abstract

This paper reports on the TREC-4 experiment carried out on the BrowsIR system developed at the University of Toronto. This is a modified version of the ST-PatTREC system that was used in TREC-3 (Charoenkitkarn et al. 1995). Six subjects participated in an experiment involving 25 topics as part of the interactive track of TREC-4.

Introduction

Our goal in this study was to gain insight into strategies used by subjects during large scale text retrieval, and the way in which those strategies are affected by different interface conditions. In particular we looked at the difference between type-in and mark-up styles of interaction. We also included a visualization of the terms used in past queries (referred to as a concept map) that some subjects had found useful in a prior study.

Research Background

The participation in TREC-4 is part of a long term research project on developing new types of user interface for information exploration. This work is motivated by the model developed by Waterworth and Chignell (1991). They argued for approaches to information exploration that combine both hypertext-like interaction and traditional text retrieval using queries. Golovchinsky (1993), and Chignell & Golovchinsky (1993) developed the QRL system as an early implementation of this approach. QRL enable users to specify queries with graphical mark-up on text giving a hypertext-like feel to the querying process. Essentially, the user pointed and clicked on the text, after which a set of matching hits was returned. Pointing and clicking on text in the QRL system was interpreted as a Boolean query. However, from the user's perspective, combinations of words in text served as hypertext anchors, with the list of returned documents acting as a menu of link endpoints.

QRL was initially a limited system that dealt with small text documents on the order of a few hundred kilobytes in size. In 1994, our group participated in the TREC-3 competition. This required us to scale up the QRL style of interaction to much larger databases of up to 500 megabytes (as required for our participation in category B of the conference). We also added in secondary resources that were expected to help people to formulate queries. These included WordNet, an online thesaurus, and an online version of the Webster's dictionary. The resulting system was called ST-PatTREC and is described by Charoenkitkarn et al (1995).

Our goal in TREC-3 was to demonstrate that graphical mark-up of queries would be a reasonable alternative to conventional methods of writing Boolean queries in command lines. The two participants who used the system to form the queries were both part of the research team and worked extensively with the system. In addition, there was no time set on how much time could be spent interacting with the system and the TREC database on each topic. In that study participants spent on average 40 minutes on each search topic.

The ST-PatTREC system did not include a lot of features normally considered to be part of text retrieval systems (including truncation, and merging of query sets). However, the results obtained with the system were in line with the other groups' results in category B (Charoenkitkarn et al. 1995). One reason for these results may be that even though the querying interface was not sophisticated, the mark-up of queries on text encouraged the users to look for the words that were actually used in the database to describe a topic. In addition, the rapid feedback that users got from viewing the hit lists returned by queries shortened the query formulation cycle and allowed users to test a wider range of query combinations within a particular period of time.

After completing our participation in TREC-3 we noted a number of deficiencies in the ST-PatTREC system. Neither Webster nor WordNet seemed to add any value during the query generation process. The subjects found that the material in the online dictionary and thesaurus was too general (not well suited to business-oriented WSJ training database), so these resources did not add value for specific queries (at least for queries on topics relevant to the Wall Street Journal database).

Users complained that fonts in ST-PatTREC were too small and hard to read. Users also found that the query history window was too small (both width and length) for long queries. Terms shown in this window also confused some users. It used a "/" to represent an OR operator and a space to represent an AND operator.

On some topics, where complex queries were needed to captures the most relevant articles, selection of the terms in the text and graphical mark-up seemed seemed insufficient. Trying to build complex queries was too difficult with the graphical querying methods. Ability to type in complex queries directly was judged to be helpful in such cases.

The searchers found that many same or similar queries had been repeated in each search session because a) query history did not keep adequate information, and b) there was no feedback to searchers concerning the overall structure of the queries that had been submitted and the corresponding results (precision and recall scores). Such feedback might have been helpful to the searchers in suggesting which query terms were better than others.

Based on the experience gained in TREC-3 and subsequent usability studies, the ST-PatTREC system was substantially modified and renamed BrowsIR. The following is a list of some of the changes that were made.

1. Type in query features were added to the system in order to allow different interface conditions to be tested (type-in versus mark-up and a hybrid combination of type-in and mark-up).

2. Based on usability results , a number of ST-PatTREC's features were removed (e.g. the workspace window, the Webster online dictionary and the WordNet thesaurus)

3. The query history window was increased in size (so that it was about six times bigger than the query history window used in ST-PatTREC).

4. A concept map view was added (as described below).

5. A set manipulation feature was added. This made it possible for users/subjects to define a set of simple queries and then merge those queries together into complex queries. Truncation was also added, so that the querying capabilities in the type-in condition roughly approximated those of a traditional text retrieval system.

Figure 1 shows components of BrowsIR.

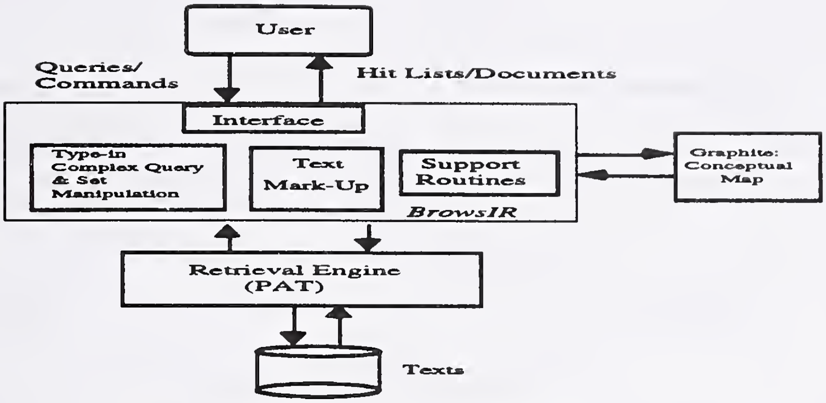


Figure 1. Components of BrowsIR

The new BrowsIR system was integrated with Graphite, a real-time graph visualization tool developed in the Department of Computer Sciences of the University of Toronto (Noik 1993). Graphite was designed and implemented to create general-purpose graph visualization utilities. The Graphite system consists of three main components: the underlying graph layout engine, a graph layout server, and an extensible Graphical User Interface (GUI) graph editor server.

The Graphite's graph layout engine can generate drawings using more than a dozen layout algorithms. The engine can compute both two- and three-dimensional layouts. In the TREC-4 experiment, however, the simpler two dimensional layout with no weighting was used. The subjects could not directly manipulate or edit the graph (i.e, the graph editor component was not used).

Graphite was used to view the overall structure of the query and frequency of query terms in relevant articles. The query graph was created in the following simple way. Queries formulated by users were simultaneously sent to Graphite. Graphite then added each query term to the graph. When two terms were related by a Boolean AND, a line linked them in the graph. Terms that appeared in multiple queries appeared only once in the map. In a session an user might specify large number of queries accumulated over the time. The query graph provided the user with an overview of queries made so far by visualizing all query terms and their relationship graphically.

In addition, there were two numbers (m,n) associated with each query term on the map. M indicated how many times the query term appeard in *n* articles judged relevant by the subjects.

Figure 2 shows an example of a query graph that was created during a search by a subject on the topic of Japanese Insider Trading (a routing topic for TREC-3).

There are ten different words used in queries tried so far. The terms *japanese*, *stock*, and *market* seems to be most used as they were linked (ANDed) with many other terms. Three articles judged were relevant by the subject at the time the concept map was generated. In those three articles, the term *japanese*, *laws*, and *market* appeared twenty two, six, and seven times respectively while neither *control* nor *regulations* appeared in those documents. Based on the available information, it seemed that the authors of the articles preferred the terminology *laws* rather than *regulations* or *control*. Also the query *japanese AND laws AND markets* looked like a good query. The other interesting query terms were *trading*, and *practices*.

Preliminary Experiment

The purpose of the preliminary experiment (carried out as part of the first author's dissertation research) was to see how people used a flexible information exploration system when faced with different types of search topics. The study sought to find out how users of dedicated mark-up and dedicated type-in systems perform when finding answers for different search topics, and if users of the hybrid system performed better than those who used dedicated mark-up and type-in systems.

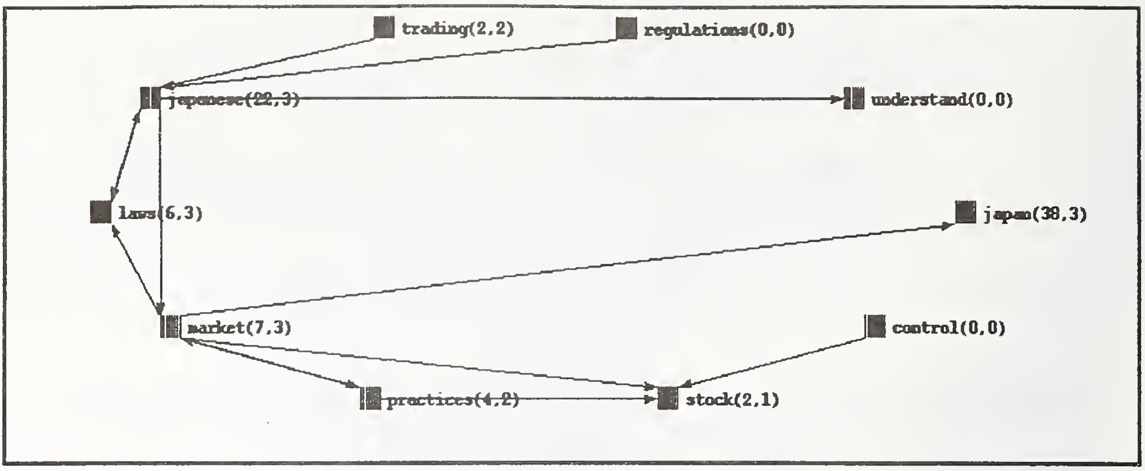


Figure 2. An example of a query graph.

Method

Thirty six subjects participated in the experiment (18 experienced searchers and 18 novices). The expert searchers consisted of four graduate students of the Faculty of Information Studies (FIS), three FIS graduates, four University of Toronto librarians, three librarians from the Toronto Metropolitan Library , two professional on-line searchers from private companies, and two information science graduates from other universities. They had various amounts of experience in conducting on-line searches using various systems. Their searching experiences varied from less than one year to eighteen years. The average amount of experience was about eight years. Most of the experts performed searching on a daily basis. Most of the novice subjects were engineering students at the University of Toronto.

The subjects were divided into three experimental groups, each of which received a different set of instructions and training on how to perform the searches. One group was trained to perform searches using only text mark-up. The second group was trained to use the type-in, and the third group was trained to use both. Each group used the same version of the BrowsIR system. However, the order of presentation of the eight search topics varied randomly from subject to subject across the entire experiment. Thus only the instructions and training differed between the three experimental groups. Within each of the experimental groups, half of the subjects were provided with the conceptual map, and half were not.

Searches were carried out on eight topics from TREC-3, classified into three factors (difficulty of terminology, number of relevant documents, and level of TREC-3 performance obtained by the University of Toronto group on those topics).

Subjects spent from three hours and twenty minutes to four hours and fifteen minutes in the experimental session, with about three and a half hours being the average session length. Subjects were run one at a time in the experiment. Each subject participated in a training session that preceded the experimental session. The training session was designed to teach subjects the skills needed to perform actual searches. The test session was divided into eight fifteen-minute search tests.

Subjects were told that their goals were to try 1) to find the first relevant document in the shortest time possible, and 2) to find as many relevant documents as possible in fifteen minutes. Each fifteen-minute search was followed by a short break during which the subject had to fill out an End of Topic Questionnaire. This questionnaire include a number of subjective ratings; satisfaction with relevance of information collected, confidence that the information retrieved was relevant, how knowledgeable they were about the topic domain, understanding of what the topic description actually meant, and how easy it was to construct a query (queries) for the topic. Subjects got no feedback from the experimenter regarding the quality of any searches, or regarding the strategies he/she employed, until after the experiment was completed. At the end of the test session, the subject was asked to fill out a background questionnaire.

Results

Overall, the average precision obtained by experts (EXPs) was higher than the precision for novices (NOVs), but the average recall of NOVs was higher than EXPs. However, both were not significantly different at the .05 alpha level (F[1,171]=2.72, p>.05 and F[1,171]=1.09, p>.25 respectively). Figure 3 shows the average precision-recall scatter graph of EXPs and NOVs.

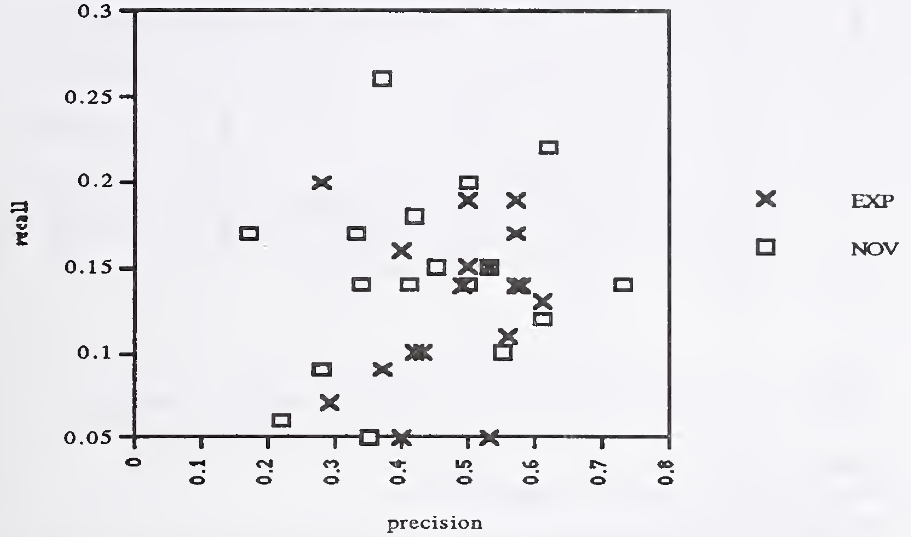


Figure 3. Scatter graph of average precision-recall of EXPs and NOVs

Cluster analysis was also carried out (Charoenkitkarn, 1996) on the data. On the basis of this analysis, the subjects were divided into two groups (recall-oriented, and precision-oriented); the majority of experts (12 of 18) were classified into the precision-oriented group, and the majority of novices (13 of 18) were classified into the recall-oriented group.

Average precision and recall scores of experts (EXP) and novices (NOV) and of recall- and precision-oriented subjects are plotted in Figure 4 along with average precision and recall of the TREC-3 experiment (reported by Charoenkitkarn et al., 1995) of the same topics using WSJ and SJMN databases. Average precision and recall of English native language and English non-native language subjects were also included as points on this plot. Figure 4 shows that subjects in TREC-3 emphasized recall. TREC-3 recall on both the WSJ and SJMN databases was much higher than for the preliminary experiment (which was carried out on the SJMN database).

In the preliminary experiment, Native English speakers were superior to non-native English speaking subjects on both precision and recall (all six subjects in the TREC-4 experiment were native English speakers). It should be noted that the goal of the subjects in the TREC-3 experiment was to come up with final queries that maximized both precision and recall. Thus documents were not selected one by one, but as a returned set based on the final query. Most importantly, there was no time limit on each search in the TREC-3 experiment. On the other hand, subjects in the preliminary experiment had only fifteen minutes to carry out each search. They had to generate queries, and then to select articles for reading before they were allowed to judge if articles were relevant or not.

The interaction between expertise (EXPERT) and search condition was marginally significant for recall (F[2,171]=4.08, p<.05) and borderline significant for precision (F[2,171]=3.06, p=.05). Most experts' performances were not affected by different search conditions. In contrast, the precision obtained by novices seems to differ between the conditions, being greatest in the mark-up condition as shown in Figure 5.

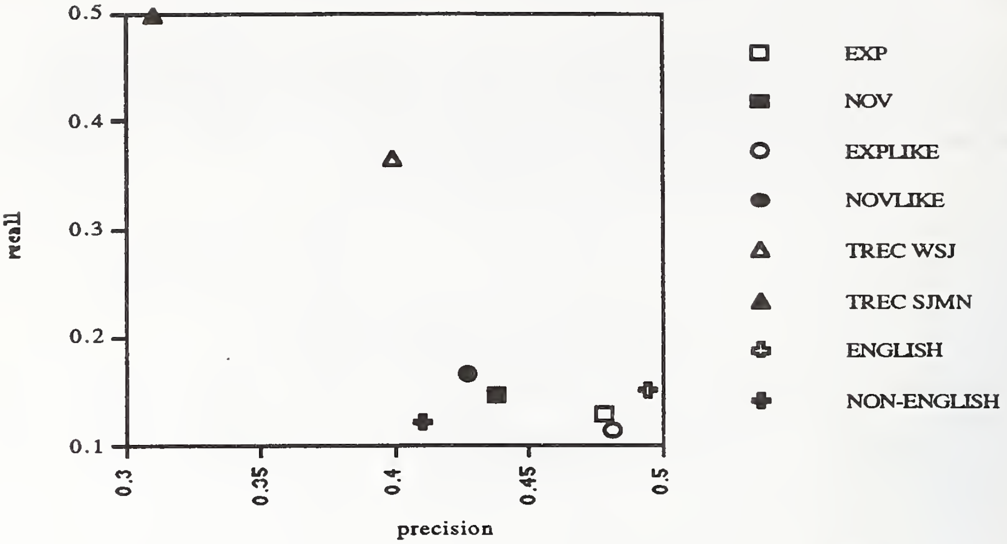


Figure 4. Average precision and recall of different groups

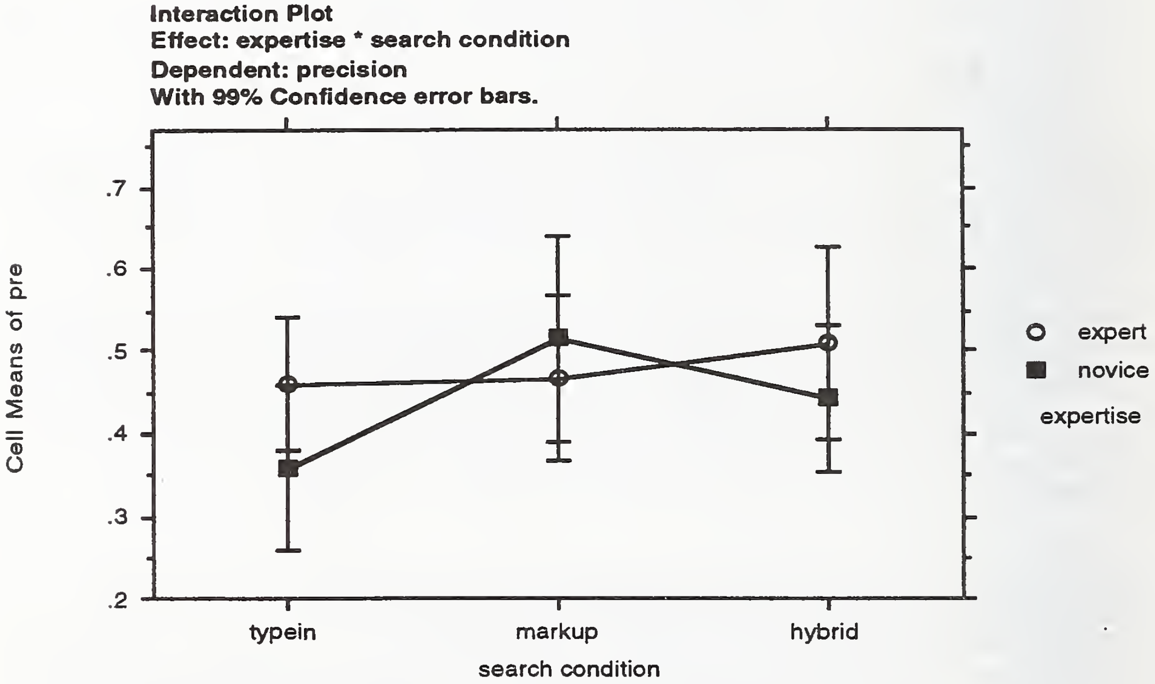


Figure 5. Precision across three search conditions of EXP and NOV

The effect on recall of the interaction between expertise and search condition is shown in Figure 6. Experts obtained roughly the same recall across all three search conditions. Recall tended to be better in the hybrid condition for novices, in contrast to the pattern for precision (Figure 5), where novices did better in the mark-up condition. One possible explanation for these results is that there was a recall-precision tradeoff for novices in the hybrid condition. According to this view, novices were more recall-oriented in the hybrid condition, leading to better recall, but worse precision. In contrast, experts in the hybrid condition tended to have better precision (than for the other two conditions) but worse recall. This raises the possibility that the flexibility of the hybrid condition (which allowed both type-in and mark-up) allowed subjects to carry out their natural strategies (i.e., recall-oriented, or precision-oriented) more fully. However, the tradeoff between recall and precision does not explain why novices tended to do worse in the type-in condition for both precision and recall.

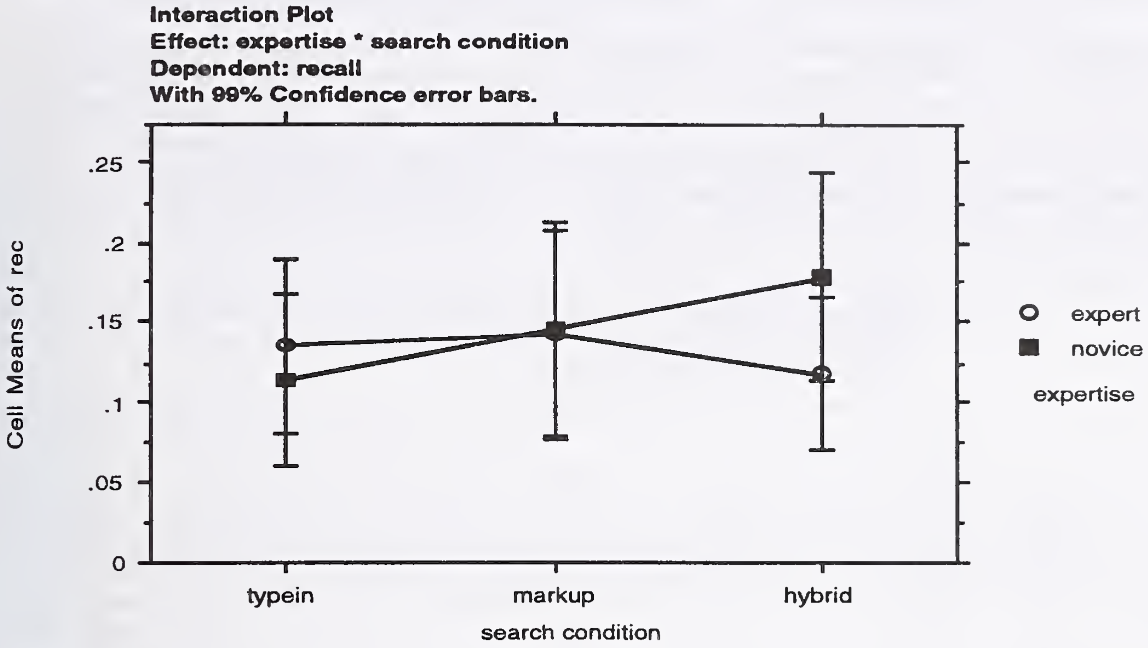


Figure 6. Recall across three search conditions of EXP and NOV

The interaction between expertise and availability of the concept map was not significant on recall, but was significant on precision (F[1,171]=11.62, p<.001). Figure 7 shows that experts got higher precision than novices in the no-map condition. However, with the map novices got higher precision than their counterparts without the map. It seems that the presence of the map helped novices to improve their precision while it led to poorer precision for experts (possibly because it distracted the attention of experts from their task, without providing them any assistance).

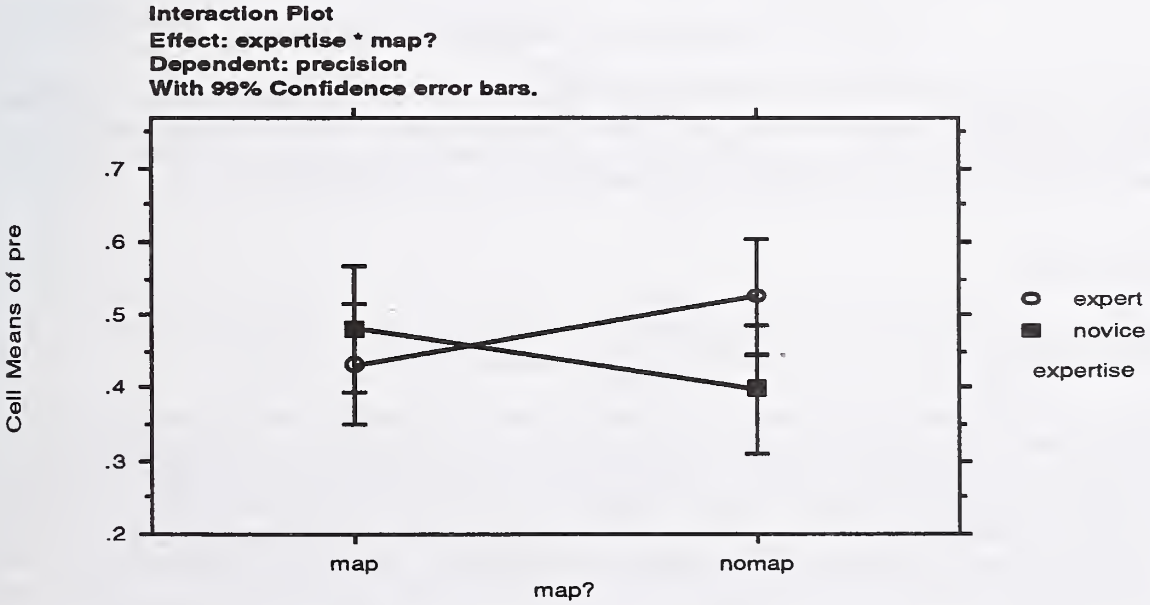


Figure 7. Precision of EXP and NOV with and without map

The interaction between expertise and map was borderline significant on time taken to find the first relevant document (TIMETAKEN) at the 95% level (F[1,171]=3.59, p<.10). When the map was present, experts spent more time to find the first relevant article and still got lower precision. Overall, the map appeared to benefit the performance of novices in this experiment while hurting the performance of experts.

Subjects filled out a questionnaire after completing each search. They made a number of subjective ratings including assessments of their satisfaction with the search and their confidence that the documents selected by them were actually relevant to the topic. The subjects' satisfaction with information retrieved was related to their precision (F[4,273]=3.653, p<.01), recall scores (F[4,273]=3.073, p<.05) and time taken to find the first relevant article (F[4,273]=7.257, p<.001). Figure 8 shows that subjects seemed to disagree strongly with the statement that they were satisfied when their recall was low, and agreed strongly that they were satisfied when their recall was high.

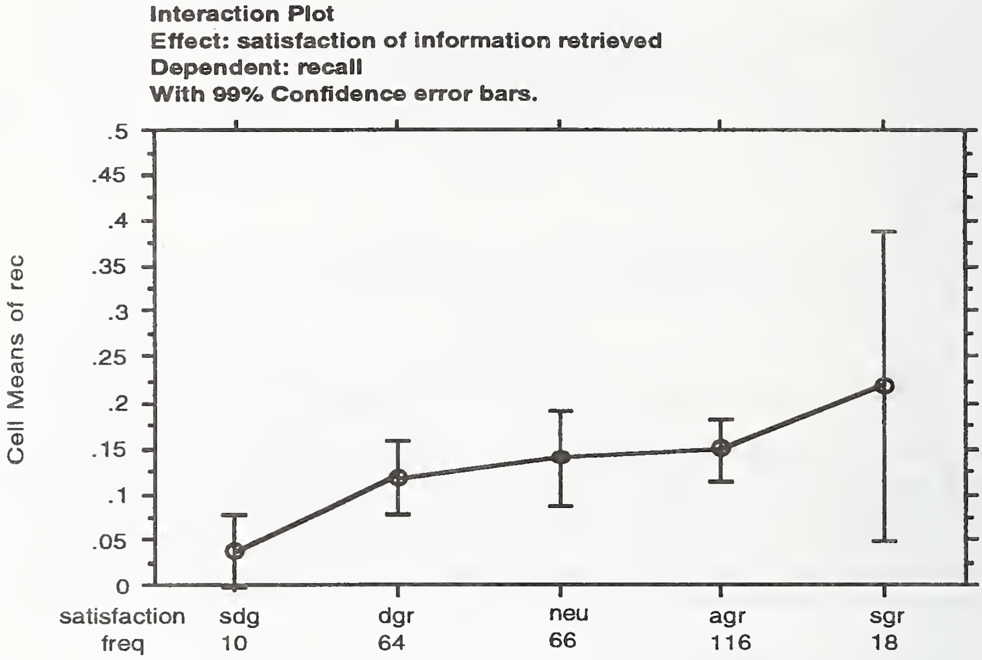


Figure 8. Relationship of recall and satisfaction

Subjects who judged themselves to be more satisfied tended to get lower precision and took less time to find the first relevant article. The result that people tended to be less satisfied when they had higher precision may be due to the fact that the experts generally expressed less satisfaction with their search results, and it was the experts who tended to be more precision-oriented and to get higher precision.

Discussion

The present discussion of this experiment provides only a brief set of findings that are most relevant to TREC-4. A more detailed account is provided in Charoenkitkarn (1996).

One major finding was that experts were more precision-oriented while novices were more recall-oriented. Overall, there was no significant difference in performance among the different interface conditions (type in, mark-up, and hybrid); however there was an interaction between expertise and interface condition, with the performance of novices being more strongly affected by differences in the types of interface used. Novices tended to do better in the markup and hybrid conditions than the type-in condition, in terms of both precision and recall. The performance of experts seemed to be much less affected by the type of interface used.

The presence of the map tended to help novices more than experts. In terms of precision, novices performed better when using the map, whereas experts got lower precision scores with the map than without it.

Satisfaction with the search was related to recall, but not to precision. In fact there was a tendency for satisfaction to be inversely related to precision (possibly a side effect of the tradeoff between recall and precision). Comparison of the recall and precision of this experiment with that of the TREC-3 experiment (Charoenkitkarn et al. 1995) showed (Figure 4) that subjects in this experiment were more precision-oriented overall.

Post-hoc analyses (t-tests) compared performance in the type-in condition with performance results combined across the mark-up and hybrid conditions. Using two-tailed tests, there was a significant difference in precision, with precision being lower in the type-in condition, but the differences between the search conditions on the other two performance variables (i.e. recall, and TIMETAKEN) were not significant. Thus precision was better with the hybrid and mark-up interfaces, without performance being any worse in terms of recall. This suggests that the improvement in precision with the mark-up and hybrid interfaces was not due to a shift in strategy towards precision at the expense of recall.

TREC-4 Experiment

The design of the experiment that we carried out for TREC-4 was constrained by the requirements of participating in the interactive track of TREC-4. In addition to carrying out these requirements using our system (BrowsIR) we were also interested in evaluating the effect of the user interface (type-in and mark-up) on performance. The results of the previous experiment (reported above) had shown that expert performance varied relatively little across the interface conditions, while novice performance was more sensitive to the interface condition used.

One issue that was studied was the relationship between articles selected for viewing and articles actually judged as being relevant. One can regard the process of selecting relevant articles as being a multistage process. First one must perform queries to find sets of articles, then one views articles that are of potential interest, finally one selects the articles that are judged to be relevant. This simple stepwise process of article selection is shown in Figure 9.

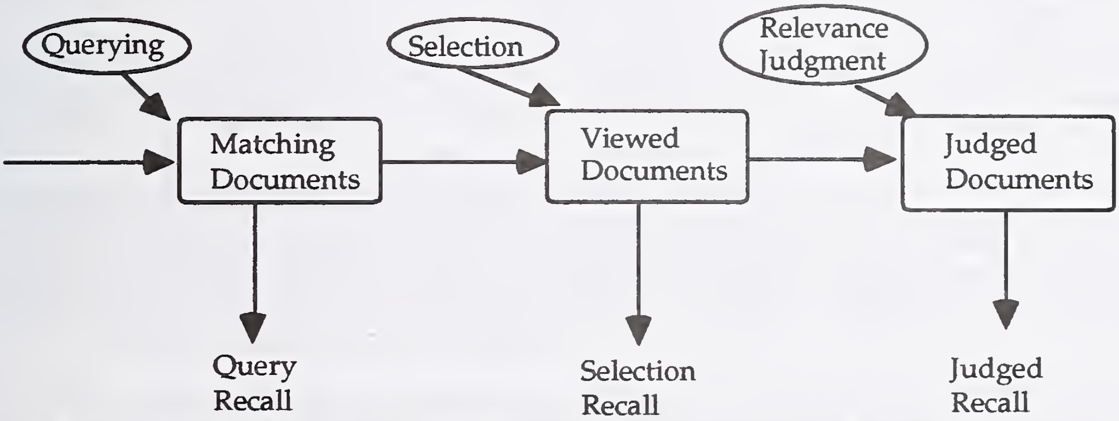


Figure 9. Different stages of Relevance Judgment and Recall Measurement.

Figure 9 shows that it is possible to measure recall at different stages of document selection. First one can measure the recall after a query has been executed. For instance, in non-interactive retrieval one typically aggregates the recall for a set of queries into a figure of merit for retrieval performance. However, in interactive retrieval the process continues, first with selection of articles to be viewed, and second with the actual judgment of which articles are relevant. Thus in the TREC-4 experiment we were interested in the relationship between selection recall and judged recall. How good would our subjects be in actually making relevance judgements after a detailed reading of articles and to what extent would the necessary loss in recall be offset by improved precision?

Method

Six subjects participated in the experiment (two experts and four novices). Participants were each paid at a rate of $10 (Cdn.) per hour for their participation in the experiment.

Each subject worked on either four or six of the TREC-4 topics as required in the specifications for participating in the interactive track (e.g., 30 minutes of search time per query). Each subject spent half of the topics using the type-in interface and half of the topics using the mark-up interface. The order in which each of these interfaces was used by each of the subjects was counterbalanced.

The experiment was run using the BrowsIR system, with the concept map available to all subjects. The experiment used the type-in and mark-up conditions. The type-in methodology simulated a traditional information retrieval system which accepted queries via keyboard. A query could consist of several terms combined with Boolean operators AND and OR. Query terms may be single (e.g. insider) or multiple-word (insider trading), or hyphenated terms (e.g. insider-trading). Truncation terms were also allowed.

In the mark-up approach, the subject could select only single word only. Multiple-word and hyphenated terms could not be selected. When additional terms were added to the query, they were ORed to the previous query term automatically. To change it to the AND, the subject had to draw a line between the two targeted terms. Truncation and set manipulations were not allowed by this approach. The main advantage of this approach was that when a selected article was displayed, the query terms were shown in the context of the article. The subject could then quickly decide to discard the article or to continue reading for more details. In the TREC-4 experiment, but not the earlier experiment described above, subjects could type in simple queries even in the mark-up condition (although they couldn't use truncation or merge query sets).

Results

Based on the R precision scores provided to us by the TREC-4 conference organizers our results were generally below the median result (for 18 of the 25 topics). This result is probably not very representative, because our subjects generally retrieved a relatively small number of articles, while the R precision measure produces better results when there is a large number of articles that are relevant and a larger number of articles is judged and ranked (in terms of relevance). When we compared our recall and precision results to other groups we found that our results were similar to those of the others. However, at least some groups appeared to have both better precision and better recall scores on average, than those obtained in our study (e.g., the group from Xerox Parc).

Table 1 shows the recall and precision obtained for each of the topics, along with the time taken to find the first relevant article. It also shows the number of relevant documents for each topic, and scores on a number of subject variables that were separately rated by participants after they completed the search on each topic. Other entries in Table 1 include the number of documents viewed that are relevant (labelled as sccrt in Table 1), and the number of judged documents that were relevant (jcrrct). The final two columns in the table refer to the precision (pre-pre) and the recall (pre-rec) associated with the set of documents that were viewed.

Student t-tests were used to compare recall and precision on selected, judged documents respectively. Since judged documents were a subset of the set of selected documents, recall for judged documents could not be greater than recall for selected documents and would generally be somewhat lower. Conversely, if judgments were effective, then precision on the judged documents should be higher than precision for the selected documents.

There was a significant different between the precision scores on selected documents (a mean of .28) and precision scores for judged documents (a mean of .55), t(24)=6.45, p<.001. There was also a significant difference between the recall scores before and after the relevance judgments, with recall dropping (as expected) from .17 to .13 (t(24)=2.32, P<.05). This result demonstrates that the relevance judgements made by subjects were generally successful identifying relevant documents, roughly doubling precision while lowering recall only moderately.

top	#rel	pre	rec	timetk	stsf	conf	und	know	quesy	judge	jcrrct	seen	scrrct	pre-pre	pre-rec
202	283	.76	.07	203	5	5	5	5	4	25	19	43	21	.49	.07
203	33	.61	.33	435	2	2	4	2	3	18	11	49	12	.24	.36
204	397	.72	.05	701	5	5	4	4	2	29	21	64	22	.34	.06
205	310	1.00	.01	353	1	5	5	3	4	2	2	26	5	.19	.02
206	47	.50	.11	338	3	4	4	2	3	10	5	54	6	.11	.13
207	74	.86	.08	116	3	5	5	4	2	7	6	21	9	.43	.12
208	54	.00	.00	.	4	3	4	2	3	12	0	43	0	.00	.00
209	87	.24	.06	400	4	4	4	3	4	21	5	39	6	.15	.07
210	57	.55	.28	73	3	3	4	3	3	29	16	63	29	.46	.51
211	323	.89	.05	82	4	3	4	3	4	19	17	39	27	.69	.08
212	153	.64	.06	251	4	4	4	3	2	14	9	82	9	.11	.06
213	21	.35	.33	542	4	4	4	3	4	20	7	34	7	.21	.33
214	5	.27	.60	90	2	3	4	3	3	11	3	36	3	.08	.60
215	183	.89	.09	165	3	4	4	3	4	18	16	26	19	.73	.10
216	36	.62	.22	144	4	3	4	3	4	13	8	36	15	.42	.42
220	24	.88	.29	65	2	4	3	3	2	8	7	38	13	.34	.54
223	363	.50	.02	314	4	4	4	4	2	16	8	78	18	.23	.05
227	347	.90	.05	161	5	5	4	3	5	20	18	52	20	.38	.06
232	9	.50	.22	172	5	5	5	3	4	4	2	41	4	.10	.44
236	43	.33	.05	1681	5	5	5	3	5	6	2	35	2	.06	.05
238	270	.43	.01	960	3	3	3	2	2	7	3	15	6	.40	.02
239	123	.47	.06	126	4	3	4	3	4	15	7	72	7	.10	.06
242	38	.64	.18	63	4	4	4	3	4	11	7	33	12	.36	.32
243	69	.16	.06	77	4	3	4	3	4	25	4	91	5	.05	.07
250	86	.00	.00	.	3	3	4	3	2	8	0	53	2	.04	.02

Table 1. Matrix of Raw Data for the TREC-4 Experiment.

Topics were divided into two categories according to the number of relevant documents that they had. The high number of relevant (HR) category contained 12 topics with 86 or more relevant documents. The low number of relevant (LR) category contained the remaining 13 topics (with fewer than 86 documents each). Figure 10 shows the relationship between precision and recall for both the viewed documents and judged documents, for topics with a high number of relevant articles. Figure 11 shows the corresponding plot for HR topics. Pre-precision and pre-recall refer to the precision and recall for the viewed documents. Recall performance is much better for topics with a low number of relevant documents (t(23)=3.54, p<.01) than for topics with a high number of relevant documents (a recall of .21, versus average recall of .04 for high number relevant topics). This is perhaps not surprising, since half an hour did not give participants long enough to find and judge a large number of documents. This result may also reflect use of a criterion where subjects want to find a "sufficient number" of documents. There was no significant difference between precision for low and high number of relevant topics (t(23)=-1.24, p>.1 However there was a tendency for precision to be greater in the HR condition (.62 versus .48).

Table 2 shows the matrix of correlations for the various measures that were collected. The correlation between selected precision and judged precision was .68. The correlation between selected recall and judged recall was .91.

There were significant (p<.05) but low (about 10% of the variance explained) correlations between satisfaction and recall for both selected (r = -.32) and judged (r = -.34) documents. This is in contrast to the results of the preliminary experiment, where there was a positive correlation between satisfaction and recall. The relationship between confidence and precision in the TREC-4 experiment was significant only for selected documents (r = .68). Thus it appears that subjects based their confidence on how the proportion of the articles that they viewed that appeared to be relevant.

Multivariate analysis of variance (MANOVA) was carried out to assess the effect of the interface condition (mark-up versus typein) on the performance variables (recall, precision, and time taken). There was no significant overall effect (F<1), nor were any of the univariate analyses of variance on the individual performance variables significant.

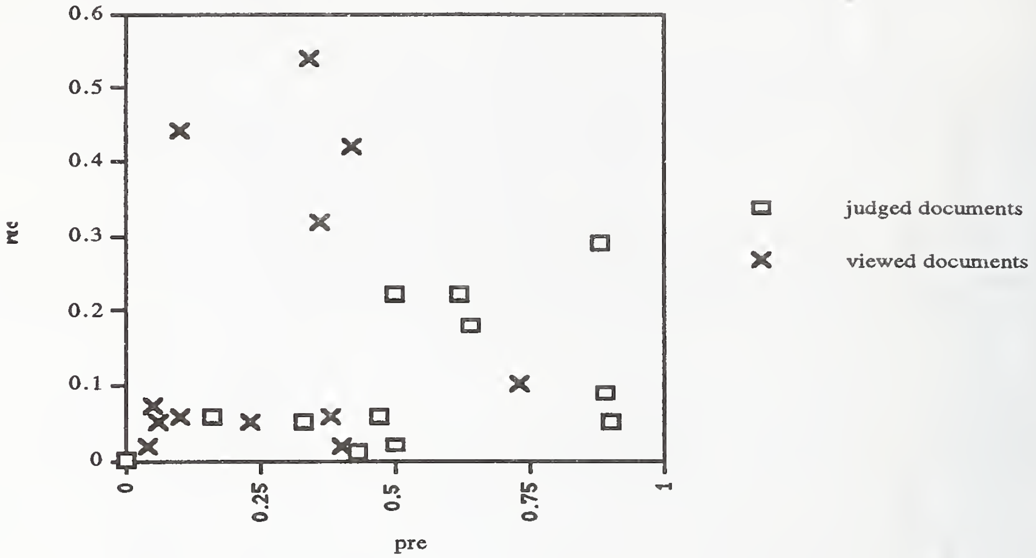


Figure 10. Relationship between Precision and Recall for Viewed and Judged Documents on HR Topics.

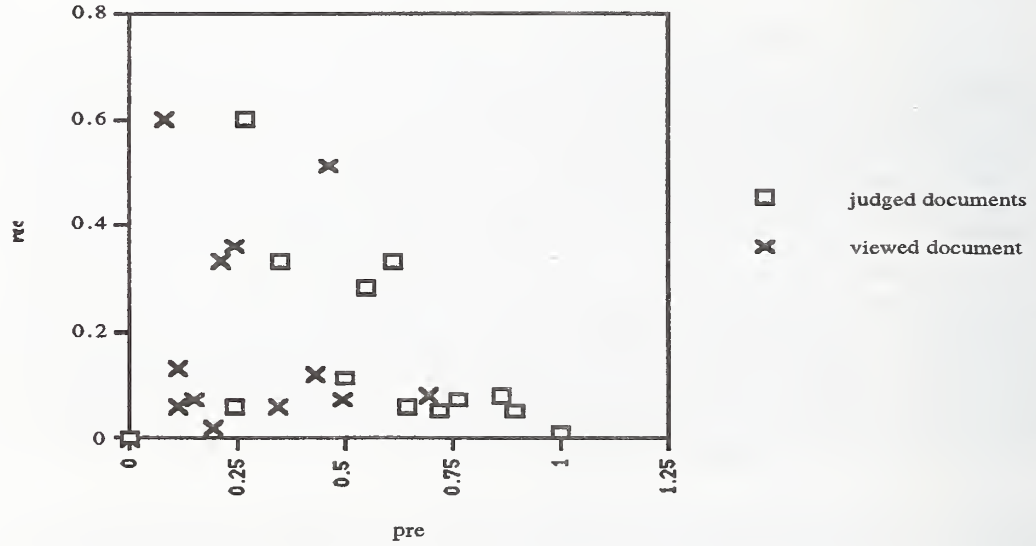


Figure 11. Relationship between Precision and Recall for Viewed and Judged Documents on LR Topics.

```
                         - -  Correlation Coefficients  - -

              PRE       PREPRE      REC       PREREC      TIME      SATISF     CONFID     QUEASY     NUMREL

PRE         1.0000      .7125     -.0337      .0595     -.2976     -.1429      .4374      .0843      .4692
            P= .       P= .000    P= .873    P= .778    P= .168    P= .495    P= .029    P= .689    P= .018

PREPRE       .7125     1.0000     -.0260      .0677     -.2600     -.0046      .0780      .0699      .4145
            P= .000    P= .       P= .902    P= .748    P= .231    P= .983    P= .711    P= .740    P= .039

REC         -.0337     -.0260     1.0000      .9188     -.2536     -.3419     -.2870     -.0070     -.5517
 .          P= .873    P= .902    P= .       P= .000    P= .243    P= .094    P= .164    P= .974    P= .004

PREREC       .0595      .0677      .9188     1.0000     -.3442     -.2952     -.2342     -.0305     -.5849
            P= .778    P= .748    P= .000    P= .       P= .108    P= .152    P= .260    P= .885    P= .002

TIME        -.2976     -.2600     -.2536     -.3442     1.0000      .2153      .2199      .0823      .0792
            P= .168    P= .231    P= .243    P= .108    P= .       P= .324    P= .313    P= .709    P= .720

SATISF      -.1429     -.0046     -.3419     -.2952      .2153     1.0000      .3608      .3591      .2029
            P= .495    P= .983    P= .094    P= .152    P= .324    P= .       P= .076    P= .078    P= .331

CONFID       .4374      .0780     -.2870     -.2342      .2199      .3608     1.0000      .2009      .3304
            P= .029    P= .711    P= .164    P= .260    P= .313    P= .076    P= .       P= .336    P= .107

QUEASY       .0843      .0699     -.0070     -.0305      .0823      .3591      .2009     1.0000     -.0664
            P= .689    P= .740    P= .974    P= .885    P= .709    P= .078    P= .336    P= .       P= .753

NUMREL       .4692      .4145     -.5517     -.5849      .0792      .2029      .3304     -.0664     1.0000
            P= .018    P= .039    P= .004    P= .002    P= .720    P= .331    P= .107    P= .753    P= .

(Coefficient / (Cases) / 2-tailed Significance)        " . " is printed if a coefficient cannot be computed
```

Table 2. Matrix of correlations between the different variables collected in the TREC-4 Experiment

Discussion

The present results showed no effect of interface condition on performance. Overall, the recall and precision scores that were obtained in our study were comparable with scores obtained by other participants in the TREC-4 interactive track.

Analysis of precision scores for selected and judged documents showed that the participants in this study were quite successful in making their relevance judgements. Precision scores were considerably higher for judged than viewed documents, while recall showed a significant, but more modest decline.

Recall performance was significantly better on topics with a low number of relevant documents in the TREC-4 experiment. This confirmed the result that was also obtained in the preliminary experiment.

Conclusions

The experiments reported here studied the effect of different user interface conditions in the context of the TREC-3 and TREC-4 tasks. In the preliminary experiment it was shown that the mark-up and hybrid conditions tending to product better performance in terms of precision than the more traditional type-in condition. When the concept map was shown to subjects, it appeared to help novices but to hurt experts (at least in terms of precision).

Charoenkitkarn (1996) provides more detailed analyses of the experiment reported here as the preliminary experiment. He found that experts tended to use more precision-oriented strategies, while novice searchers tended to use recall-oriented strategies.

The TREC-4 experiment conformed to the requirements of the interactive track. Subjects had longer (30 minutes vs. 15 minutes in the preliminary experiment) to search on each topic. Recall and precision scores were obtained that were generally consistent with typical results obtained by other participants in the interactive track of TREC-4. Our comparison of recall and precision for selected (viewed) versus judged articles showed that the relevance judgements increased precision greatly, while reducing recall moderately.

Recall performance was generally better on topics that had a low number of relevant documents. However, there was also a tendency for precision to be better on topics with a high number of relevant documents.

The relationships between the subjective and objective performance variables was somewhat inconsistent between the two experiments reported here. In the preliminary experiment, satisfaction increased with higher levels of recall (but not precision) whereas in the TREC-4 experiment there was no relationship between satisfaction and either recall or precision. In contrast, there was a significant relationship between confidence and precision in the TREC-4 experiment, but not in the preliminary experiment.

It is unclear why there is not a strong relationship between objective search performance measures such as precision and recall, and subjective ratings such as satisfaction and confidence. We think that it is probably wrong to dismiss the subjective ratings as unreliable and focus only on the objective measures. Ellis, Furner-Hines, and Willett (1994) distinguished among different types of relevance assessment (judge-relevant, searcher-relevant, navigator-relevant). If there are in fact different types of relevance, then it may well be that subjective ratings of performance are responding to different types of relevance than the "official" view of relevance as defined by expert judges. However, this cannot be the whole story, since the results of our TREC-4 experiment show that the relevance judgements made by our subjects led to a marked improvement in precision scores.

Acknowledgements

This research was funded by a grant from the Information Technology Research Centre of Excellence of Ontario. We would like to thank Manny Noik for making his graph layout software available for use in our experiment.

References

Charoenkitkarn, N. (1996). *Searching Behaviour.* Unpublished Ph.D Dissertation. Department of Industrial Engineering, University of Toronto, Toronto, Canada.

Charoenkitkarn, N. Chignell, M.H., and Golovchinsky, G. (1995). Interactive Exploration as a Formal Text Retrieval Method: How Well can Interactivity Compensate for Unsophisticated Retrieval Algorithms. *Proceedings of the ThirdText REtrieval Conference (TREC-3).* D.K. Harman (Ed.). pages 179-199. National Institute of Standards and Technology Special Publication 500-225. Gaithersburg, Maryland, 1995.

Ellis, D., Furner-Hines, J. & Willett, P. (1994). On the measurement of inter-linker consistency and retrieval effectiveness in hypertext databases. In: Croft, W.B. & van Rijsbergen, C.J. (editors) *Proceedings of the 17th International Conference on Research and Development in Information Retrieval.* pp. 51-60. London: Springer Verlag.

Golovchinsky, G. (1993). *Queries-R-Links: Query-based browsing in a full-text retrieval system.* Unpublished Ma.Sc. Thesis, Department of Industrial Engineering, University of Toronto, Toronto, Canada.

Golovchinsky, G. and Chignell, J.A. (1993). Queries-R-Links: Graphical markup for text navigation. Proceedings of INTERCHI '93, 454-460. N.Y.: ACM Press.

Noik, E.G. Exploring Large Hyperdocuments: Fisheye Views of Nested Networks. pages 192-205. *Hypertext 93*, Seattle, USA., November 1993.

Waterworth, J.A. and Chignell, M.H. (1991). A Model of Information Exploration. Hypermedia, 3(1)35-58.

Appendix: Addendum to System Description for Interactive Experiments

I. System description

1. Screen dump of "typical" screen.

The functions of the BrowsIR system that were used in our study are summarized in Figures 12 and 13. Figure 12 shows a snapshot of an interaction in the mark-up user interface condition, while Figure 13 shows a snapshot of an interaction in the type-in user interface condition.

rigure 12 shows several queries were tried and reported in the query history along with their corresponding results. The figure shows an AND operator marked up on the text as a line connecting the three selected terms.

Figure 13 shows a similar interface, but with the query typed in the margin instead of marked up on the text itself. Other minor differences between the two conditions (as can be seen by comparing Figures 12 and 13) occur in the windows on the right hand side of the screen. The differences include a smaller query history window in the mark-up condition as well as other windows that differed slightly in size between the two condition.

2. Usable features of the interface.

In the mark-up approach, the subject could enter only a single term. Multiple-word and hyphenated terms could not be selected. When the second term or later term was added to the query, it could be ORed to the previous query term automatically. To change it to the AND, the subject had to draw a line between the two targeted terms. Truncation and set manipulations were not allowed in this approach. The main benefit of this approach occurred when a selected article was displayed. In this case the screen presentation showed where the query term(s) was located on the article. The subject could then decide quickly whether or not to to discard the article, or to continue reading for more details instead.

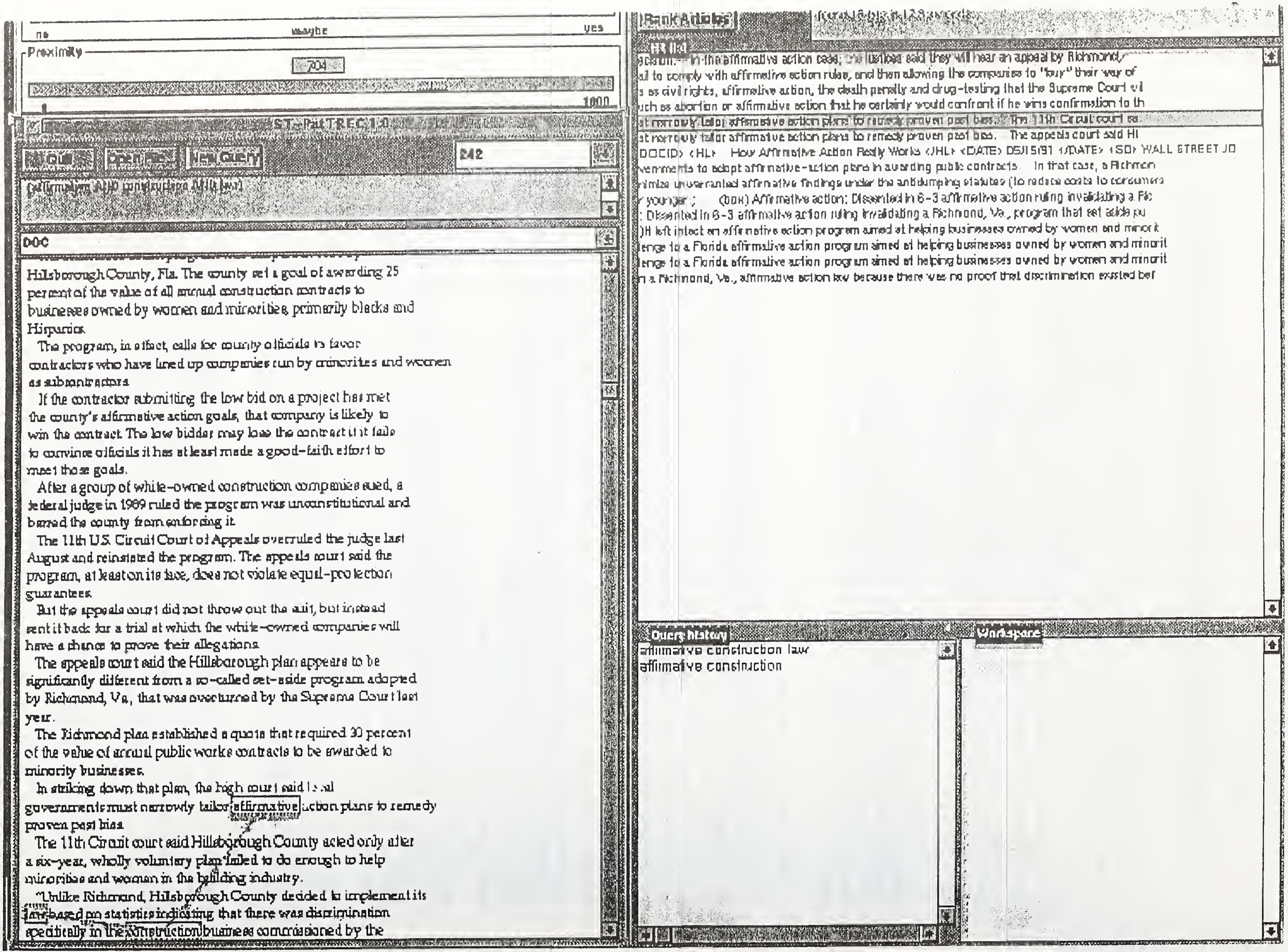
The type-in interface (Figure 13) simulated a traditional information retrieval systems which accepts queries via keyboard (typically entered as command strings). A query could consist of several terms combined with the Boolean operators AND and OR. Query terms could be single words (e.g. insider) or multiple-word phrases (e.g., insider trading), or hyphenated terms (e.g. insider-trading). Truncation terms were also allowed. The search engine searched case-insensitively. For instance, the query term "*insider trading*" or "*Insider Trading*" was considered the same.

BrowsIR used disjunctive normal form Boolean querying. It gave higher priority to the Boolean AND operator than to the OR operator. Thus the query "*japan* AND insider trading OR japan* AND practices* " was be interpreted as "*(japan* AND insider trading) OR (japan* AND practices)*. An asterisk (*) represented a wild card. E.g. *japan** matched any terms that had "j-a-p-a-n" as the first five letters. Thus japan* would match to words such as *japan* or *japanese.*

Like many Boolean-based retrieval systems, the type-in approach allowed subjects to compare and combine sets of previous results (stored in order in the Query History window). The subject could further refine search by comparing or combining sets. The subject could compare sets to find the different or common elements. These operations were the same as the mathematical set operations: *union*, *difference* and *intersection.*

Combining sets of results into one was advantageous when subjects were interested in merging the results from different queries. When combining sets, duplications were eliminated. The second set manipulation found the difference between one set and another. The difference could be defined as any articles in the first set that were not in the second. The last (intersection) set manipulation found the matches common to both sets. It let subject determine if two sets shared any results.

Both the type-in and mark-up versions of the system provided a concept map. This was a two-dimensional presentation of a network that was constructed based on the terms that had been used in previous queries.



(affirmative AND construction AND law)

Hillsborough County, Fla. The county set a goal of awarding 25 percent of the value of all annual construction contracts to businesses owned by women and minorities, primarily blacks and Hispanics.

The program, in effect, calls for county officials to favor contractors who have lined up companies run by minorities and women as subcontractors.

If the contractor submitting the low bid on a project has met the county's affirmative action goals, that company is likely to win the contract. The low bidder may lose the contract if it fails to convince officials it has at least made a good-faith effort to meet those goals.

After a group of white-owned construction companies sued, a federal judge in 1989 ruled the program was unconstitutional and barred the county from enforcing it.

The 11th U.S. Circuit Court of Appeals overruled the judge last August and reinstated the program. The appeals court said the program, at least on its face, does not violate equal-protection guarantees.

But the appeals court did not throw out the suit, but instead sent it back for a trial at which the white-owned companies will have a chance to prove their allegations.

The appeals court said the Hillsborough plan appears to be significantly different from a so-called set-aside program adopted by Richmond, Va., that was overturned by the Supreme Court last year.

The Richmond plan established a quota that required 30 percent of the value of annual public works contracts to be awarded to minority businesses.

In striking down that plan, the high court said local governments must narrowly tailor affirmative action plans to remedy proven past bias.

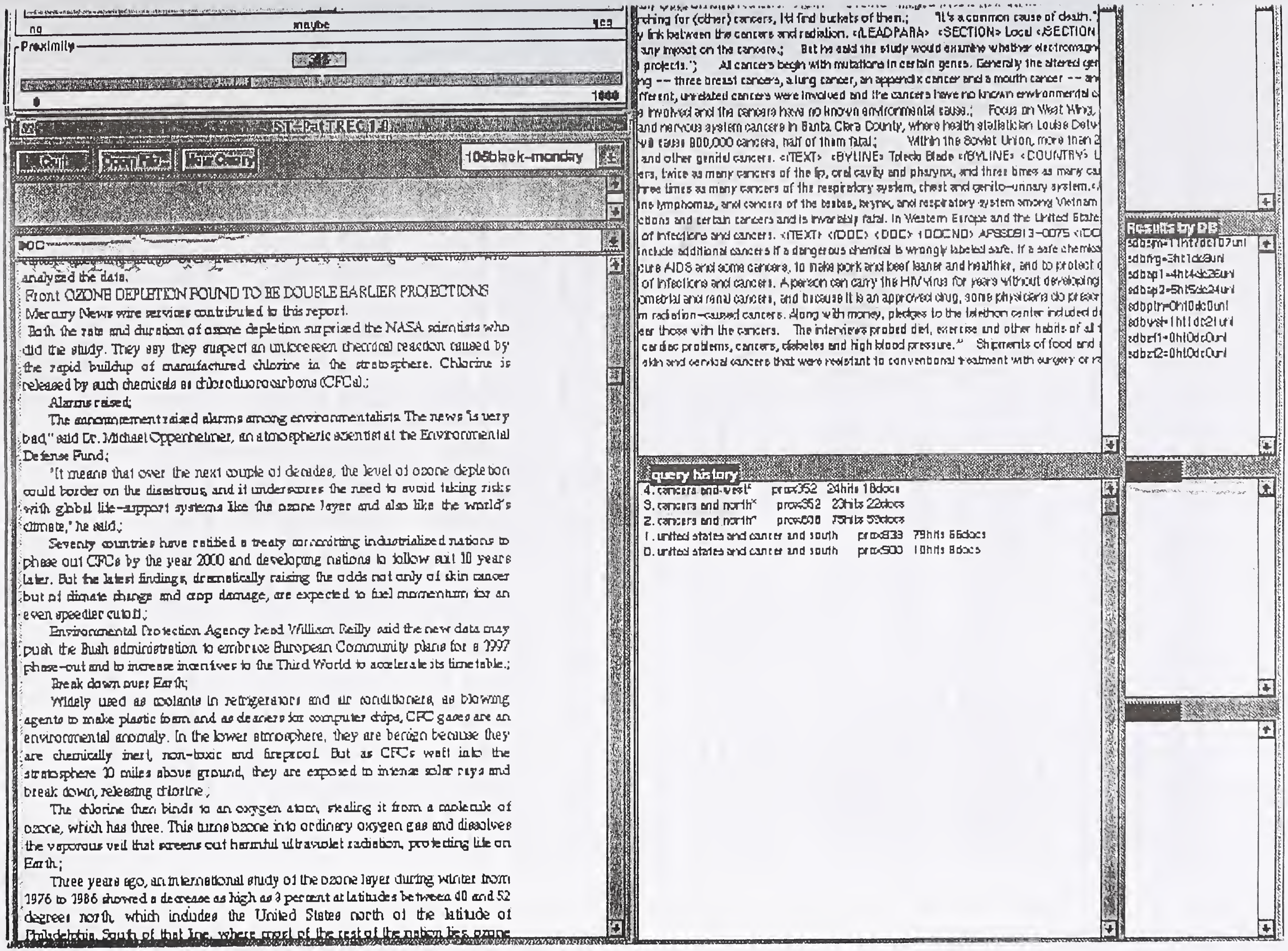
The 11th Circuit court said Hillsborough County acted only after a six-year, wholly voluntary plan failed to do enough to help minorities and women in the building industry.

"Unlike Richmond, Hillsborough County decided to implement its law based on statistics indicating that there was discrimination specifically in the construction business commissioned by the

ssion. In the affirmative action case, the justices said they will hear an appeal by Richmond,
al to comply with affirmative action rules, and then allowing the companies to "buy" their way of
s as civil rights, affirmative action, the death penalty and drug-testing that the Supreme Court wil
uch as abortion or affirmative action that he certainly would confront if he wins confirmation to th
st narrowly tailor affirmative action plans to remedy proven past bias. The 11th Circuit court sa
st narrowly tailor affirmative action plans to remedy proven past bias. The appeals court said Hi
DOCID> <HL> How Affirmative Action Really Works </HL> <DATE> 05/15/91 </DATE> <SO> WALL STREET JO
vernments to adopt affirmative-action plans in awarding public contracts. In that case, a Richmon
nimize unwarranted affirmative findings under the antidumping statutes (to reduce costs to consumers
r younger ; (box) Affirmative action: Dissented in 6-3 affirmative action ruling invalidating a Flo
: Dissented in 6-3 affirmative action ruling invalidating a Richmond, Va., program that set aside pu
)H left intact an affirmative action program aimed at helping businesses owned by women and minorit
lenge to a Florida affirmative action program aimed at helping businesses owned by women and minorit
lenge to a Florida affirmative action program aimed at helping businesses owned by women and minorit
n a Richmond, Va., affirmative action law because there was no proof that discrimination existed bef

affirmative construction law
affirmative construction

Figure 12. A screen display of a search using mark-up methodology.



analyzed the data.

Front OZONE DEPLETION FOUND TO BE DOUBLE EARLIER PROJECTIONS

Mercury News wire services contributed to this report.

Both the rate and duration of ozone depletion surprized the NASA scientists who did the study. They say they suspect an unforeseen chemical reaction caused by the rapid buildup of manufactured chlorine in the stratosphere. Chlorine is released by such chemicals as chlorofluorocarbons (CFCs).;

Alarms raised;

The announcement raised alarms among environmentalists. The news "is very bad," said Dr. Michael Oppenheimer, an atmospheric scientist at the Environmental Defense Fund;

"It means that over the next couple of decades, the level of ozone depletion could border on the disastrous, and it underscores the need to avoid taking risks with global life-support systems like the ozone layer and also like the world's climate," he said.;

Seventy countries have ratified a treaty committing industrialized nations to phase out CFCs by the year 2000 and developing nations to follow suit 10 years later. But the latest findings, dramatically raising the odds not only of skin cancer but of climate change and crop damage, are expected to fuel momentum for an even speedier cutoff.;

Environmental Protection Agency head William Reilly said the new data may push the Bush administration to embrace European Community plans for a 1997 phase-out and to increase incentives to the Third World to accelerate its timetable.;

Break down over Earth;

Widely used as coolants in refrigerators and air conditioners, as blowing agents to make plastic foam and as cleaners for computer chips, CFC gases are an environmental anomaly. In the lower atmosphere, they are benign because they are chemically inert, non-toxic and fireproof. But as CFCs waft into the stratosphere 10 miles above ground, they are exposed to intense solar rays and break down, releasing chlorine.;

The chlorine then binds to an oxygen atom, stealing it from a molecule of ozone, which has three. This turns ozone into ordinary oxygen gas and dissolves the vaporous veil that screens out harmful ultraviolet radiation, protecting life on Earth.;

Three years ago, an international study of the ozone layer during winter from 1976 to 1986 showed a decrease as high as 3 percent at latitudes between 40 and 52 degrees north, which includes the United States north of the latitude of Philadelphia. South of that line, where most of the rest of the nation lies, ozone

rching for (other) cancers, I'd find buckets of them.; "It's a common cause of death."
y link between the cancers and radiation. </LEADPARA> <SECTION> Local </SECTION
any impact on the cancers.; But he said the study would examine whether electromagn
projects."; All cancers begin with mutations in certain genes. Generally the altered ge
ng -- three breast cancers, a lung cancer, an appendix cancer and a mouth cancer -- an
fferent, unrelated cancers were involved and the cancers have no known environmental c
e involved and the cancers have no known environmental cause.; Focus on West Wing,
and nervous system cancers in Santa Clara County, where health statistician Louise Delv
will cause 800,000 cancers, half of them fatal; Within the Soviet Union, more than 2
and other genital cancers. </TEXT> <BYLINE> Toledo Blade </BYLINE> <COUNTRY> U
ers, twice as many cancers of the lip, oral cavity and pharynx, and three times as many ca
hree times as many cancers of the respiratory system, chest and genito-urinary system.<
ine lymphomas, and cancers of the testes, larynx, and respiratory system among Vietnam
ctions and certain cancers and is invariably fatal. In Western Europe and the United State
of infections and cancers. </TEXT> </DOC> <DOC> <DOCNO> AP890813-0075 </DO
include additional cancers if a dangerous chemical is wrongly labeled safe. If a safe chemis
cure AIDS and some cancers, to make pork and beef leaner and healthier, and to protect
of infections and cancers. A person can carry the HIV virus for years without developing
ometrial and renal cancers, and because it is an approved drug, some physicians do presc
m radiation-caused cancers. Along with money, pledges to the telethon center included d
ker those with the cancers. The interviews probed diet, exercise and other habits of all
cardiac problems, cancers, diabetes and high blood pressure." Shipments of food and
skin and cervical cancers that were resistant to conventional treatment with surgery or ra

query history

4. cancers and west* prox352 24hits 18docs
3. cancers and north* prox352 23hits 22docs
2. cancers and north* prox600 75hits 59docs
1. united states and cancer and south prox833 79hits 66docs
0. united states and cancer and south prox500 10hits 8docs

Results by DB

sdbsjm=11ht7dc107unl
sdbfrg=3ht1dc3unl
sdbap1=4ht4dc35unl
sdbap2=5ht5dc24unl
sdbptn=0ht0dc0unl
sdbwsj=1ht1dc21unl
sdbef1=0ht0dc0unl
sdbz2=0ht0dc0unl

Figure 13. A screen display of a search using type-in methodology.

Edges in the network represented AND relations that had been used in the queries. Further explanation of the concept map is provided in the body of this paper.

3. Style of interface: *gui*

The user interface was a multipane window (as shown in Figures 12 and 13) constructed using ObjectWorks SmallTalk on a Sparc-20.

II. Experimental conditions

1. Searcher characteristics.
   a. Number of searchers in experiment: *6*
   b. Number of searchers per topic: *1*
   c. Age/age group of searchers.
 average of 30 years old (estimated). Ages ranged from 23 to 35.
   d. IR searching experience of searchers.
   [average of 5 years.
   e. Educational level of searchers.
   1 Ph.D., 2 Ph.D. students, 1 M.Sc., and 2 M.Sc. students.
   f. Undergraduate major of searchers.
   [4 engineering, and 2 arts majors.
   g. Experience/familiarity with subject of topic.
   none of the searrchers were domain experts in any of the topics that they searched on.
   h. Work affiliation of searchers.
   4 students , one consultant working in the Financial services industry, and one Library school graduate who is currently working for IBM but who also has an interest in an academic position.

2. Task description

Instructions to Subjects

This experiment will take approximately three hours. Rest breaks will be scheduled as needed. You will be carrying out searches on five topics. Each of the five searches will take 30 minutes.

The session will being with a brief introduction to the information retrieval system you will be using. This will be followed by a short period of training. You will then participate in the test session. You will perform interactive searches for five topics (each lasting up to 30 minutes) without further help. At the beginning of each search you will be handed a topic description. You will then be asked to rate the difficult of the terminology used in the topic description. You can then start the search using any of the system features that you were trained on.

For each search, find as many documents as you can which address the given information problem, but without finding too many irrelevant documents. You should complete the task in around 30 minutes. Your performance will be measured by the number of relevant documents that you find, and the proportion of documents found that are actually relevant.

Each search will be followed by a break during which you will fill out a short questionnaire regarding the particular search that just completed (your familiarity with the topic, the ease of query construction, your confidence in the quality of the relevance documents etc.).

You will get no feedback from the experimenter regarding the quality of the previous search or regarding the strategies you employ until after the experiment is completed. After the last search and questionnaire, you will be asked to fill out a background questionnaire.

3. Training
   a. Description of the training process

All six of the subjects in the TREC-4 experiment had already used the BrowsIR system in the preliminary experiment. Thus they had already received that training. In addition, they received an additional 10 minutes or so training in the TREC-4 experiment to refresh their memory of how the system worked. The following paragraphs describe the training that they had received earlier as part of their participation in the preliminary experiment. On average there was about a two month break between when a subject participated in the preliminary experiment and when that same subject participated in the TREC-4 experiment.

Description of Training for the Preliminary Experiment

Each subject participated in a training session that preceded the experimental session. The training session was designed to teach subjects the skills needed to perform actual searches. The decision to use one hour of training was based on the results of earlier pilot testing with two pilot subjects. Initially only half an hour was planned for the training session, but this time was found to be insufficient.

At the beginning of the training session, the experimenter explained the procedure to the subject. The experimenter then showed the subject how to use the system and conduct searches step-by-step by following the instructions in a training manual that was constructed especially for the experiment (the training manual is shown in Appendix B). After the experimenter finished the demonstration, the subject had about thirty minutes to explore the BrowsIR system on his/her own. During this period, the experimenter provided help if necessary or when it was requested. After finish the training session, the subject proceeded to the test session.

The test session was divided into eight fifteen-minute search tests. During each test, the subject was handed a search topic (the order of search topics was randomized so that each subject worked with a different random order). The subject was free to consult the manual and whatever notes he/she might have taken during the training session. Only three of the thirty six subjects ever consulted their manuals. Although searches on each topic were limited to 15 minutes, subjects typically took more than two hours to finish all eight topics because of the need for rest breaks, and due to system crashes that occurred from time to time (these crashes or system failures were not frequent, but resulted from the fact that a research prototype, rather than commercial software, was being used to run the experiment).

The subjects were told that their goals were to try 1) to find the first relevant document in the shortest time as possible and 2) to find as many relevant documents as possible in fifteen minutes. Each fifteen-minute search was followed by a short break during which the subject had to fill out an End of Topic Questionnaire as explained in the "Method of data recording and information recorded" section below. The subject got no feedback from the experimenter regarding the quality of any searches, or regarding the strategies he/she employed, until after the experiment was completed. At the end of the test session, the subject was asked to fill out a Background Questionnaire as explained in the following section.

b. Time for training, in minutes.
Mean: *10 minutes. All subjects previously participated in anthoer three-hour experiment using a similar search system.*
Range: 5-20 minutes
Note: Subjects already had about an hour of training on the BrowsIR system as part of their participation in the preliminary experiment. However, some subjects had not received training on the concept map in the preliminary and thus the training that they received in the TREC-4 experiment had to cover this gap in their knowledge of the system.

III. Search process

1. Clock time
Mean: *No final query was required for searchers. All subjects spent 30 minutes for each search.*
Median: *30 minutes*
SD: *0 minute*
Range: *30 minutes*

2. Number of documents "viewed" during the search.
a. Definitions of viewing:

In the BrowsIR system, subjects could view a set of hits that matched a query in terms of a list of one line excerpts from each article. (Some of the excerpts on different lines came from the same article because the query could be matched to different portions of some articles, so that the multiple hits or excerpts would match to each of these different portions of the article. Clicking on one of these "hits" then called up the full text of the corresponding article in another window. For the purposes of our participation in the interactive track of TREC-4, a document was assumed to have been "viewed" once it was called up in the text window by clicking on the corresponding hit (excerpt) in the window of hits that matched the current query.

b. Number of items viewed per search

Mean: *46*
Median: *41*
SD: *21*
Range: *15-91*

3. Number of iterations per search.
a. Definition of iteration:

The concept of an iteration is somewhat vague in BrowsIR. For the type-in condition, a new query corresponds to a new query command (i.e., a new query line). However, for mark-up, queries are entered incrementally and it is generally difficult to say where one query ends and another begins. Thus we adopted the heuristic that a new query was signalled whenever the subject decided to view a hit using a query that was different from the query that was in operation when the previous hit was selected.

b. Number of iterations per search.
Mean: *6*
Median: *5*
SD: *3*
Range: *2-16*

4. Number of terms used in queries.
a. Number of terms in the first query of a search, per search.
Mean: *3*
Median: *4*
SD: *3*
Range: *2-8*
b. Number of terms in the final query of a search, per search.
Mean: *4*
Median: *5*
SD: *3*
Range: *2-10*

5. Use of system features.
Proximity operations: *mean 1, median 1, range 0-6*
Merged queries: *mean 1, median 1, range 0-3*
Truncation: *mean 1, median 0, range 0-6*

6. Number of user errors made per search.
a. Definition of an error:

It was not possible to make an error in the mark-up condition. Thus there were no errors for mark-up. In type-in, errors represented illegal Boolean commands or syntax.

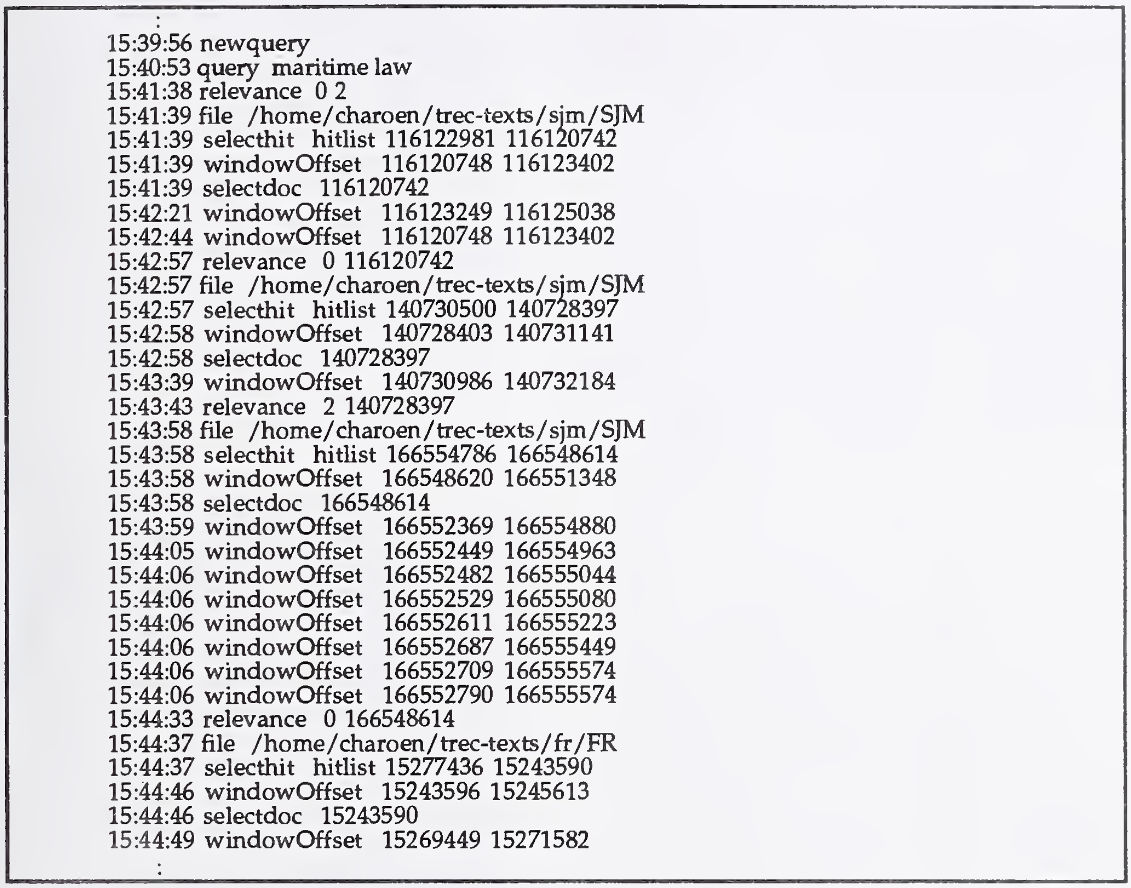
b. Number of user errors per search.

The experiment noted some errors in the type-in condition for the preliminary experiment, but could not remember any errors for the type-in condition in the TREC-4 experiment.

Mean: *0*
Median: *etc.*
SD:
Range:

7. Search narrative for topic 236.

Figure 14 below shows an excerpt for a transaction log from one of the subjects in the type-in condition of the TREC-4 experiment. The first query used was "maritime law". He clicked on the relevance button which had no effect in this case. He then looked at one of the hits that was returned based on the query. He then moved to different portions of the document (different offsets) by clicking on the scrollbar. He then indicated that he didn't think the document was relevant (i.e., he assigned it a neutral 0 rating). He then selected another document from the hitlist. He then changed the viewing position in the document once. After that he selected that document as being relevant (a relevance rating of 2 was positive, since the scale ranged from -2 to +2). He then selected another article and looked through it in detail (hence the large number of lines detailing the changes in the offset position). He then gave the document a relevance rating of 0 ("neutral"). After that he selected one more document and then changed the viewing position again, at which point this excerpt of the transcript terminates.



```
      :
15:39:56 newquery
15:40:53 query  maritime law
15:41:38 relevance  0 2
15:41:39 file  /home/charoen/trec-texts/sjm/SJM
15:41:39 selecthit  hitlist 116122981 116120742
15:41:39 windowOffset  116120748 116123402
15:41:39 selectdoc  116120742
15:42:21 windowOffset  116123249 116125038
15:42:44 windowOffset  116120748 116123402
15:42:57 relevance  0 116120742
15:42:57 file  /home/charoen/trec-texts/sjm/SJM
15:42:57 selecthit  hitlist 140730500 140728397
15:42:58 windowOffset  140728403 140731141
15:42:58 selectdoc  140728397
15:43:39 windowOffset  140730986 140732184
15:43:43 relevance  2 140728397
15:43:58 file  /home/charoen/trec-texts/sjm/SJM
15:43:58 selecthit  hitlist 166554786 166548614
15:43:58 windowOffset  166548620 166551348
15:43:58 selectdoc  166548614
15:43:59 windowOffset  166552369 166554880
15:44:05 windowOffset  166552449 166554963
15:44:06 windowOffset  166552482 166555044
15:44:06 windowOffset  166552529 166555080
15:44:06 windowOffset  166552611 166555223
15:44:06 windowOffset  166552687 166555449
15:44:06 windowOffset  166552709 166555574
15:44:06 windowOffset  166552790 166555574
15:44:33 relevance  0 166548614
15:44:37 file  /home/charoen/trec-texts/fr/FR
15:44:37 selecthit  hitlist 15277436 15243590
15:44:46 windowOffset  15243596 15245613
15:44:46 selectdoc  15243590
15:44:49 windowOffset  15269449 15271582
      :
```

Highlighting Relevant Passages for Users of the Interactive SPIDER Retrieval System

Daniel Knaus, Elke Mittendorf, Peter Schäuble, Páraic Sheridan
(knaus|mittendorf|schauble|sheridan)@inf.ethz.ch
Swiss Federal Institute of Technology (ETH)
CH-8092 Zürich, Switzerland

Abstract

We report here on our participation in the 1995 Text Retrieval (TREC 4) conference. This was limited to participation in the interactive searching task due to a re-implementation of our SPIDER retrieval system. We report on two aspects of the SPIDER system that are particularly useful for interactive searching, namely the use of fast query evaluation, and the use of passage highlighting in presenting retrieved documents to users. We also report on the lessons learned from observing people interact with our system and note the challenge faced by researchers wishing to evaluate the interactive use of information retrieval systems.

1 Introduction

Due to the evolution of the SPIDER system over the years, in part because of lessons learned from participation in past TREC conferences, and to allow a more seamless integration of SPIDER's different facets, we have this year undertaken a major redesign and re-implementation of the SPIDER retrieval system. A major objective of this redesign was to modularise the system based on client-server architecture so as to increase its flexibility and allow it to be easily adapted to the different types of tasks for which we use it—running our TREC experiments on the one hand, while also using it as a vehicle for our continuing research into multilingual and multi-media information retrieval. Because of our participation this year in the interactive track of TREC, we also took the opportunity to provide new search and document inspection functionality, like identifying and highlighting relevant passages of documents chosen for viewing.

Although we are now reaping the fruits of our extended redesign effort, we did not have it completed with sufficient time to load the TREC collection and perform the adhoc and routing tasks. This led us to concentrate on the interactive track for which we successfully completed the primary task. We therefore concentrate here on reporting the aspects of the SPIDER system that are specifically directed toward interactive searching, namely efficient query evaluation in section 2, and passage identification in section 3. The system functionality is summarized in section 4. We also report in section 5 on our experiences in performing the interactive task.

2 Fast Query Evaluation

For interactive searches we want to have efficient and effective retrieval, i.e.: *Find as many documents as possible relevant to the topic, without too much rubbish, in about 30 minutes.* We decided to use the best weighting scheme of last year's TREC as our basic method, namely the BM25(2.0, 0, ∞, 0.75) weighting of the Okapi team [Robertson et al., 1994].

$$RSV(q, d_j) = \textstyle\sum_{\varphi_i \in q \cap d_j} a_{ij} b_i,$$

where

$$a_{ij} = \frac{\mathrm{ff}(\varphi_i, d_j)}{2(1/4 + 3/4 \cdot l_j/\Delta) + \mathrm{ff}(\varphi_i, d_j)},$$

$$b_i = \log\left(\frac{1/2 + n - \mathrm{df}(\varphi_i)}{1/2 + \mathrm{df}(\varphi_i)}\right) \cdot \mathrm{ff}(\varphi_i, q),$$

$\mathrm{df}(\varphi_i)$ = document frequency,

$\mathrm{ff}(\varphi_i, d_j)$ = feature (term) frequency,

l_j = number of tokens in the document d_j,

Δ = average of l_j over all documents,

n = number of documents in the collection.

Fast query evaluation is achieved by using an inverted file with the posting lists sorted by decreasing feature frequencies:

$$\varphi_i \mapsto \mathrm{df}(\varphi_i), \{(\mathrm{ff}(\varphi_i, d_j), d_j) | \mathrm{ff}(\varphi_i, d_j) > 0\}$$

Only the important parts of the posting lists (high feature frequency postings) are processed, less important parts are skipped. Furthermore, the document lengths are cached in main memory for faster access. We originally wanted to use the pruning algorithm of Persin [Persin, 1994] but like other speedup approaches

[Buckley & Lewit, 1985] this approach does not take into account the number of documents to be ranked, i.e. no matter how many documents are to be retrieved, these algorithms try to optimize the overall retrieval effectiveness.

Because there is in fact a tradeoff between the speed of query evaluation and the effectiveness achieved, we wanted to be able to tune our system to balance between fast and effective query evaluation. Although the system should guarantee a certain response time, we assume that users will accept slightly longer response times if they submit longer queries or ask for more documents to be returned in the ranked list. It was with this in mind that we formed our approach to efficient query evaluation.

If a query q were to be evaluated completely then $k_{tot} = \sum_{\varphi_i \in q} \text{df}(\varphi_i)$ postings would have to be processed. We want to process only k postings ($k \leq k_{tot}$) such that the query is evaluated within a certain time (restricted number of disk accesses) while still ensuring that the list of the r top ranked documents is not significantly different from the beginning of the list produced by processing all k_{tot} postings. The response time is proportional to the number of postings we have to process, and the quality of the ranking produced is then dependant on the ratio k/k_{tot} and the number of requested documents, r. In our implementation we used the following simple formula to estimate k:

$$k = c \cdot |\vec{q}\,| \cdot r \tag{1}$$

The constant c is used to tune the system: a small c allows fast but less effective retrieval and with a large c the query evaluation takes more time but the ranked list is better.

The k postings are selected from the k_{tot} postings such that the ranked list is most effective. The algorithm simultaneously works on the posting lists of all query features and processes as the next posting that which adds the highest estimated partial similarity $S_{\text{nxt}}(\varphi_i)$ to a score. $S_{\text{nxt}}(\varphi_i)$ is estimated from the last posting of φ_i that has been processed. Because the postings are ordered by decreasing $\text{ff}(\varphi_i, d_j)$ we know that the feature frequency of the next posting must be equal or less than the current one. The length of the document of the next posting is estimated by the average document length (Δ).

The following pseudo-code illustrates our algorithm:

```
1. estimate $k$ by (1)

2. initialization

     for each $\varphi_i \in q$ with $\text{df}(\varphi_i) > 0$ do
       open cursor on the posting list of $\varphi_i$
       compute $b_i$
       set $S_{\text{nxt}}(\varphi_i)$ to $b_i \cdot a_i^{\max}$
     endfor
     create a structure of accumulators
     set $\varphi_h = \text{argmax}\{S_{\text{nxt}}(\varphi_i) | \varphi_i \in q, \text{df}(\varphi_i) > 0\}$

3. accumulation

     while ($k > 0$) and ($S_{\text{nxt}}(\varphi_h) > 0$) do
       get next posting $(d_j, \text{ff}(\varphi_h, d_j))$ of $\varphi_h$
       if found then
         increment accumulator $A_j$ by $b_h \cdot a_{hj}$
         set $S_{\text{nxt}}(\varphi_h)$ to $b_h \cdot a_{hj}^{\text{nxt}}$
       else
         set $S_{\text{nxt}}(\varphi_h)$ to 0
       endif
       set $\varphi_h = \text{argmax}\{S_{\text{nxt}}(\varphi_i) | \varphi_i \in q, \text{df}(\varphi_i) > 0\}$
       decrement $k$
     endwhile

4. identify $r$ accumulators with highest values
```

Using the the current posting $(d_j, \text{ff}(\varphi_i, d_j))$ we estimate the document weight a_{ik} of the next posting $(d_k, \text{ff}(\varphi_i, d_k))$ by

$$a_{ij}^{\text{nxt}} = \frac{\text{ff}(\varphi_i, d_j)}{2 + \text{ff}(\varphi_i, d_j)}$$

assuming that the document length of the next document d_k is Δ. For estimating a_{ik} before having processed any postings we use the upper bound

$$a_i^{\max} = 1.$$

For features φ_i with $\text{df}(\varphi_i) = 0$ the estimated partial similarity $S_{\text{nxt}}(\varphi_i)$ is initialized by 0.

Accesses to the inverted list of postings of a feature are cached of course, i.e. it is not necessary to access the disk each time another posting is to be read.

With the parameter c it is possible to choose between fast but less effective retrieval or effective but slower retrieval, whichever is preferred. It is even possible to change c between two query evaluations. This can be very helpful to quickly get an idea of the data and then to set up the query more carefully and evaluate it more effectively.

With respect to r, the estimation formula (1) is not yet realistic because the constant c does not have the same effect for low and high values of r. The influence of c on the retrieval effectiveness is subject to further investigations.

3 Passage Retrieval

In previous experiments [Knaus et al., 1994, Mittendorf & Schäuble, 1994] we have shown that Hidden Markov Models (HMM) provide a natural and effective method for retrieving relevant passages from documents. In this section we summarize the ideas of

HMM-based passage retrieval and we describe the refinements and changes we have made in the passage retrieval method in contrast to previous papers. Relevant passages of a document are identified and highlighted when the text of documents are presented to the user in the interactive search environment to help speed up decisions about the relevance or non-relevance of the document. Users can also mark relevant passages for relevance feedback, rather than using the features of the whole document. We use Hidden Markov Models to identify relevant passages as follows:

We assume that the document was produced by a HMM. In other words, a document is considered as a sequence of basic units produced by a finite state automaton, e.g. a sequence of tokens (occurrences of indexing features such as Porter-reduced non-stop-words [Porter, 1980]) or a sequence of sentences. The production of basic units in each state and the transitions outgoing from each state can be described with probability distributions. These probability distributions can be optimized on training material by applying the so-called Baum–Welch [Baum, 1972] training algorithm. As in other description oriented approaches for probabilistic retrieval [Fuhr, 1992], to get probability distributions which are tractable given a query, we map each basic unit to a value which describes the similarity between the basic unit and the query; for example if a single indexing feature is the basic unit then the similarity is described by $\mathrm{ff}(\varphi_i, q) \cdot \mathrm{nidf}^2(\varphi_i)$ [Knaus et al., 1994], where the normalized inverse document frequency is defined by $\mathrm{nidf}(\varphi_i) := 1 - \frac{\log(\mathrm{df}(\varphi_i)+1)}{\log(n+1)}$. We assume that there are certain states of a HMM which model the production of the passages that are relevant to the query and there are other states that are responsible for producing non-relevant passages. Given a document and a query and the corresponding sequence of similarity values, the Viterbi-algorithm [Viterbi, 1967] then enables us to find the most probable state sequence in the Hidden Markov Model which has produced this document. We identify as interesting passages those passages which are produced in the states of the HMM which are responsible for producing relevant passages.

We have recently adapted this approach of relevant passage identification to be more suitable to the interactive environment in which it is now used:

From Token to Sentence Descriptions:

Experiments have shown that it is difficult to provide meaningful similarity descriptions between a single token and a query. We therefore sought to find larger units to provide a more meaningful unit for measuring similarity with the query, but which retain the property that a sequence of such units still contains structure which can be modeled by a HMM. Sentences fulfill these requirements and so have been chosen as our basic units. Very rudimentary heuristics have been used to decide sentence boundaries in our texts. A sentence is simply considered to be a string s consisting of at least 15 characters such that the terminal character $s[N]$ must be a typical end-of-sentence delimiter like '?', '!' or '.'. To cater for certain types of abbreviation, we specify that if the sentence is terminated by a period ($s[N]$ ='.') then $s[N-2]$ must be an alphanumeric character (e.g. to avoid false recognition of an end-of-sentence in "John F. Kennedy"). To compute a similarity value, the sentence s_k and the query q are preprocessed as usual by removing stop-words and performing Porter-reduction on the remaining words. A similarity value between s_k and q is then computed with ff · nidf-weighting and the scalar product.

$$\mathrm{sim}(q, s_k) := \sum_{\varphi_i \in q, s_k} \mathrm{ff}(\varphi_i, q) \cdot \mathrm{nidf}^2(\varphi_i) \cdot \mathrm{ff}(\varphi_i, s_k).$$

We have not yet made thorough evaluations to investigate whether sentential units lead to better effectiveness than token units. In a few experiments that we have conducted, it performed as well as token based retrieval. However, the use of sentential units does lead to better *efficiency*, so even for very long documents the passage retrieval is performed in a short time.

Continuous Hidden Markov Models:

Changing from discrete Hidden Markov Models to continuous Hidden Markov Models was mainly due to implementational reasons. We wanted to use the very flexible Hidden Markov Toolkit (HTK) [Young et al., 1993], which, at that time, supported only continuous Hidden Markov Models.

Different Model Layouts:

We experimented with different model-layouts for the Hidden Markov Models. A relevance model R and an irrelevance model I consist of only one state. Probability distributions were trained with a random selection of relevant or irrelevant query-document pairs from queries 101–200 and documents from the AP1 and AP2 collections.

We tried various different combinations of the relevance HMM R and the irrelevance HMM I in the Hidden Markov Model that models the document production:

- If documents are assumed to have exactly one relevant passage we take the model:

  $$[I]R[I]$$

  We adopt the notation for regular expressions from the HTK notation [Young et al., 1993]. The expression $[X]$ denotes an optional traversal of the model X. (Note: This model—though very restrictive—yielded the best results when using passage level evidence in the routing or ad-hoc task for data from last year's TREC. We do not report the experiments here.)

- If we allow the document to have an arbitrary order of relevant and irrelevant documents we can choose the model
$$[I]\{R[I]\}$$
where $\{X\}$ denotes an arbitrary number of traversals, but at least one. It is distracting for some users that this model sometimes yields two or more consecutive relevant passages.

- We thus chose the following model for interactive searches
$$[I]\{RI\}[R]|IR|R$$
where $(X|Y)$ denotes the alternative of the two models X and Y.

4 System Functionality

The SPIDER System [Schäuble, 1993] is based on a client-server architecture. The retrieval server's tasks are kept to a minimum (inserting and deleting document descriptions, evaluating indexed queries), and a retrieval client (in the case of TREC-4) is used to provide the following search and inspection functionality:

- Evaluating queries: the queries are indexed by the client and then sent to the server. The result returned by the server is a ranked list consisting of document identifiers, retrieval status values and titles of the ranked documents.

- Inspecting documents: the documents are loaded from disk and passed to the passage retrieval component which is part of the client. The document is then displayed to the user with relevant passages highlighted for attention.

- Relevance feedback: The user can specify passages or whole documents as relevant. Features of relevant passages are stored and are added to the query when relevance feedback is performed. To limit the size of feedback queries, a maximum of the 50 highest weighted terms from the feedback query are considered.

Users presently interact with the retrieval client through a text-based user interface. This allows them to specify queries and view the ranked list of resulting documents, view the full texts of documents with passages judged relevant by the system appropriately marked, specify full documents as either relevant or nonrelevant, or explicitly record marked passages as relevant, and perform relevance feedback so that the query is expanded with features from relevant passages (not documents) and resubmitted for evaluation. The full set of functions available to the users of our system are included in Appendix A.

5 Interactive Searching

For our participation in the interactive track we had 11 different people search for documents relevant to the TREC topics set aside for the interactive task. For 13 of the 25 interactive topics there was only one person who searchered for documents on that topic, ten of the topics were tried by two people, and in two cases three different searchers tried to find documents for a given topic. Of the 11 people who performed the interactive task for us, only three would consider English to be their native language, so the task of formulating queries for the TREC topics and skimming retrieved documents to judge relevance was difficult for many of the participants. (For example, many of our participants did not know what *"affirmative action"* was). The process of interactive searching that these searchers engaged in has essentially three parts, coinciding with the three classes of functionality of the system outlined in the previous section:

Query Specification: Searchers started out with the text of the TREC topic and were encouraged during training to specify a query that only included the important words or phrases from this description. They were also encouraged to add other related words to the query specification - to "express in as many ways as possible what you're looking for". Query specification also encompasses those points during an interactive session where a user feels that a dead-end has been reached on a given query formulation so a completely new query is given. The result of query evaluation was a ranked list of documents (doc-ids and titles) with their associated rank positions and RSV values. This list contained a default of 20 documents, but the user was free to adjust the number of documents displayed.

Viewing Retrieved Documents: Once a query had been evaluated the users would generally spend some time browsing the list of titles and would then choose to view the full text of documents that were of interest. The full text of a chosen document would be presented to the user, but passages of the text that were judged to be interesting by our system were marked and users were encouraged during training to direct their attention to these marked sections.

Judging Relevance: Based on either the title of a document in the ranked list, or on the full text of a viewed document, a user could categorise the document as either relevant or non-relevant. Users were aware that it was much more important to find and mark relevant documents rather than non-relevant ones. If the full text of a document had been viewed, then the user could also explicitly mark highlighted passages as relevant. Only these marked passages were used in relevance feedback, not whole documents. Users were encouraged during training to perform relevance feedback as soon as they had one or two relevant passages.

The users therefore proceeded by iterating through these actions, using either relevance feedback or specifying a new query once they had browsed the full list of documents retrieved from an initial query. From observing the interactive sessions, we have concluded that the most effective search strategy seemed to involve specifying as precise a query as possible initially, with the objective of finding at least one or two documents with good relevant passages so that these relevant passages could be used through relevance feedback to retrieve more relevant material. In general, relevance feedback on passages appeared to work very well. Initial query formulation was therefore critical, as the most difficult task seemed to be finding that first good relevant document with which to launch feedback. Our starting advice on formulating long initial queries may therefore have gone against us, long initial queries seemed to retrieve documents that, although relevant to the topic in general, or rather to parts of the topic, were not directly relevant to the information need expressed. For example, when searching for information on *how affirmative action has affected the construction industry*, it was quite easy to find documents relating to affirmative action, and also easy to find documents reporting on the construction industry, but it was very difficult to find documents tying the two together.

In general, users were happy with the performance of the system if they got the chance to use relevance feedback, as this proved to be a very effective way of finding relevant documents once one or two relevant passages had been discovered. On the other hand, users found it extremely frustrating if they were unable to find any good relevant documents (i.e. with passages that they could mark relevant) with which to launch relevance feedback, as they generally found that query specification was the most difficult part of their task and wanted to avoid having to re-formulate queries after unsuccessful searches.

6 Conclusions

This was our first year of participation in the interactive track of TREC, and we feel it was a very valuable and rewarding experience - to watch different people use our system in trying to find documents relevant to given query topics, especially when the wording of the query topic and the SPIDER system together seemed to conspire to really test the searcher's patience. Our experiences with observing the interactive sessions left us very satisfied with the effectiveness of our fast query evaluation and passage retrieval reported on here. The fast query evaluation resulted in user queries being evaluated in a matter of two or three seconds, though the longer queries submitted through relevance feedback took longer than this. We also found that passage identification and highlighting was an efficient and very effective tool, especially for concentrating users' attention to the most important parts of documents. Given that passage identification was performed when a user chose to view the full text of a document, documents could still be presented to the user with highlighted passages within a matter of seconds. We were particularly pleased with the effectiveness of our decision to allow users to distinguish between documents that were relevant in general and specific passages that were relevant, and to use only user-marked relevant passages for feedback.

We note however, the lack of a definitive means of evaluating our system, for example in order to compare it to other systems that have been used in the interactive track of TREC 4. This lack of systematic evaluation is a result of both our time constraints due to our re-implementation efforts, and also a lack of a coherent evaluation plan for the interactive track of TREC. This is an evolving track and one which must address some very complex evaluation issues. We look forward to playing our role in this evolution.

If we were to choose a direction for further improvement in our interactive system it would certainly be to address the formulation and evaluation of initial queries. We hope to investigate in particular the use of query expansion techniques (e.g. [Qiu & Frei, 1993]), but this will go hand in hand with a detailed examination of the tradeoffs between efficiency and effectiveness resulting from our work described in section 2, so that we can be confident that our expanded queries will be evaluated efficiently while retaining an acceptable level of effectiveness.

Acknowledgements: The authors would like to thank Roger Aeppli and Philipp Häfliger for their help with implementation. We also want to thank all participants of the interactive track for taking part, namely Jean-Paul Ballerini, Sushil Bhattacharjee, Tore Bratvold, Dana Davis, Ulrich Gerloff, Oliver Lorenz, Arlette Piquet, Yonggang Qiu, Marc Roth, David Tonhofer, and Martin Wechsler.

References

[Baum, 1972] Baum, L. E. (1972). An Inequality and Associated Maximization Technique in Statistical Estimation of Probabilistic Functions of Markov Processes. *Inequalities*, 3,1–8.

[Buckley & Lewit, 1985] Buckley, C., & Lewit, A. F. (1985). Optimization of Inverted Vector Searches. In *ACM SIGIR Conference on R&D in Information Retrieval*, pp. 97–110.

[Fuhr, 1992] Fuhr, N. (1992). Probabilistic Models in Information Retrieval. *The Computer Journal*, 35(3),243–255.

[Knaus et al., 1994] Knaus, D., Mittendorf, E., & Schäuble, P. (1994). Improving a Basic Retrieval Method by Links and Passage Level Evidence. In *TREC-3 Proceedings*, pp. 241–246.

[Mittendorf & Schäuble, 1994] Mittendorf, E., & Schäuble, P. (1994). Document and Passage Retrieval Based on Hidden Markov Models. In *ACM SIGIR Conference on R&D in Information Retrieval*, pp. 318–327.

[Persin, 1994] Persin, M. (1994). Document Filtering for Fast Ranking. In *ACM SIGIR Conference on R&D in Information Retrieval*, pp. 339–348.

[Porter, 1980] Porter, M. F. (1980). An Algorithm for Suffix Stripping. *Program*, 14(3),130–137.

[Qiu & Frei, 1993] Qiu, Y., & Frei, H. (1993). Concept Based Query Expansion. In *ACM SIGIR Conference on R&D in Information Retrieval*, pp. 160–169.

[Robertson et al., 1994] Robertson, S. E., Walker, S., Jones, S., Hancock-Beaulieu, M., & Gatford, M. (1994). Okapi at TREC-3. In *TREC-3 Proceedings*.

[Schäuble, 1993] Schäuble, P. (1993). SPIDER: A Multiuser Information Retrieval System for Semistructured and Dynamic Data. In *ACM SIGIR Conference on R&D in Information Retrieval*, pp. 318–327.

[Viterbi, 1967] Viterbi, A. J. (1967). Error bounds for convolutional codes and an asymptotically optimal decoding algorithm. *IEEE Transactions on Information Theory*, 13(2),260–269.

[Young et al., 1993] Young, S., Woodland, P., & Byrne, W. (1993). *HTK Version 1.5: User, Reference & Programmer Manual.* Entropic Cambridge Research Laboratory, Sheraton House, Castle Park, Cambridge CB3 OAX, England.

A Addendum to System Description for Interactive Experiments

A.1 System Description

Usable Features

- *New-Topic:* Begin a session searching for documents for a given TREC topic. The topic number is entered and the text of the topic is displayed on the screen.
- *Query:* Enter a query to the system. The current (previous) query (initially blank) is displayed to remind the user, but a completely new query must be entered. Any relevance judgements made in the current session are *retained* however; to begin with a completey clean slate, the command *New-Topic* must be given.
- *Display-Last-Query:* The text of the current (previous) query is displayed.
- *Feedback:* Perform relevance feedback using features from passages that have been judged relevant by the user. The list of marked relevant passages is maintained from the *New-Topic* command. The user can remove passages previously marked relevant by marking the document non-relevant.
- *Ranked-List:* Display the current ranked list of documents returned from the current (previous) query. The list includes the rank position, RSV value, Doc-id and document title. Documents are also marked as follows:
  - **V** The full text of the document has been viewed.
  - **R** The whole document has been marked as relevant.
  - **RP** A passage has been marked relevant.
  - **N** The whole document has been marked non-relevant.
- *More-Titles:* Display *X* documents in the ranked list. The default value is 20 documents.
- *Show-Document:* Present the full text of the chosen document. The document is first passed through the passage identification module so that interesting passages can be highlighted when the text is presented.
- *Mark-Relevant-Passage:* Mark a highlighted passage as relevant. The marked passage is noted for future relevance feedback and the whole document is marked relevant for the purpose of the interactive task.
- *Relevant-Doc:* Mark a document as relevant. This does not contribute to future feedback commands, but the document is saved for the interactive results.
- *Non-Relevant-Doc:* Mark a document as not relevant. Unless this is being used to reverse early relevance judgements (where a document or passage was earlier marked relevant), this has no real effect, other than to provide more information for analysis by us.
- *Exit:* End the session and leave the SPIDER system.

Style of Interface We used a text-based command interface for our experiments. It was sufficient to use the first two characters of the commands given above.

A.2 Experimental Conditions

Searcher Characteristics

We did not record detailed biographical information for each searcher. Their main characteristics are as follows:

- *Number of Searchers* 13 in total performed searches, the results of 11 different searchers were used in our final submission.
- *Number per Topic* Ranged from 1 to 3.
- *Searcher Ages*
  - (25 to 29) 9
  - (30 to 34) 1
  - (35 to 39) 1

- *Search Experience* None of our participants could be described as "professional" searchers, but rather as casual users of information retrieval systems. Although we had one librarian take part, none of her sessions were included in the submitted results (we submitted for each topic the searcher who marked most documents relevant). Many of our participants have some experience with information retrieval from a research perspective. One was a complete novice searcher, but was chosen because she was American and so had an edge on background knowledge over the European searchers.
- *Educational Level* All hold at least a Masters Degree or equivalent (Diplom), 3 also hold PhD degrees.
- *Undergrad Major*
  - 7 Computer Science,
  - 1 Engineering,
  - 1 Mathematics,
  - 2 Business.
- *Topic Familiarity*
  - 9 Minimal familiarity.
  - 2 Some background knowledge.
- *Work Affiliation*
  - 8 CS Dept. ETH Zurich,
  - 2 Union Bank of Switzerland (also computer people),
  - 1 Institue of Industrial Engineering and Management, ETH.

Task Description

**Find as many documents as possible relevant
to the topic, without too much rubbish,
in about 30 minutes**

The document collection consists of about 760,000 documents. Documents vary greatly in size, from some very short documents, to some very long documents.

The SPIDER system marks individual passages in documents that are found to be relevant. When reading a retrieved document, it is only necessary to look at those passages marked as relevant. Relevant passages are marked with "*1" (the first relevant passage).

It is not useful to specify *negative* search information, as one might use in a boolean system. Instead, try to focus your query more toward the exact topic, using additional search terms that will favour relevant documents.

Try to find as many ways as possible to express the query concept, for example by specifying a list of synonyms and related terms, rather than using a small set of search terms.

The SPIDER system has a *feedback* option that uses *passages* that you have marked as relevant to find other similar relevant documents. Once you have found two or three documents with relevant passages, the "feedback" command is an option to find more relevant documents.

Searching Tips

Spend some time planning your initial query. Your first query should be designed to retrieve a few relevant documents so that you can use these to focus your query and find more relevant documents.

- Use only those terms from the topic description that are central to the meaning: so terms like *"Find documents that..."* will not help.
- Understand the concept/idea of the topic and try to find related terms to add to your query.

Once you have a ranked list of documents, identify the relevant documents and use these to find others...

- Note the terms used in relevant documents. See if any of these terms might be added to your query. Resubmit a new query including the terms that you have found in good documents.
- Mark relevant passages as relevant and use them to perform "feedback". Note, only documents with *passages* marked as relevant are used for feedback (these are displayed as "RP" in the ranked list).

Training

The training process consisted of allowing the searchers time to read through the task description and the searching tips presented above, after which any questions or clarifications were dealt with. The searchers were then given a practice topic with which to practice. This was one of the official TREC interactive topics, but one that would not be given to that searcher during the actual task. We observed the searchers during their practice session and provided suggestions and guidance where we thought necessary.

The time taken to train each searcher was not accurately recorded. I estimate that the average time is somewhere around 45 minutes.

A.3 Search Process

Clock Time

(This was recorded in Minutes - Seconds are given here)

Mean	Median	SD	Range
1862.4	1860	218.732	1440 - 2280

Documents Viewed

We consider *viewing* a document as seeing the text of a document. We compiled the numbers for this by counting the number of times the command *show-document* was issued during a session. (If the same document was viewed twice, then it counts twice)

Mean	Median	SD	Range
31.76	29	9.479	20 - 54

Iterations per Search

We consider as an *iteration* either the submission of a query by the searcher, or the use of feedback. We present here results based only on the total number of iterations per session. Individual results for the use of the *Query* and *Feedback* commands may be found later in this section.

Mean	Median	SD	Range
7.36	6	5.971	2 - 30

Terms per Query

We consider here only those queries entered by the searcher (we don't in fact have data about the queries submitted through relevance feedback). Since we did not perform the secondary interactive task, there was no real *final query* as such, so we present here only figures computed over all queries, and then over all first queries.

Mean	Median	SD	Range
5.375	4	3.353	1 - 18

Terms in First Query

Mean	Median	SD	Range
6.36	5	2.659	3 - 13

Use of Features

Note that with our text-based user interface, it was necessary to issue the *Ranked-List* command to view the current ranked list of documents each time the full text of a document had been viewed so the ranked list was lost from the screen. This accounts for the high usage of this command.

Command	Mean	Median	SD	Range
Query	5.68	5	5.821	1 - 28
Display-Last-Query	0.48	0	1.357	0 - 5
Feedback	1.68	2	0.802	0 - 3
Ranked-List	19.96	21	5.086	7 - 27
More-Titles	0.68	1	0.69	0 - 2
Show-Document	31.76	29	9.479	20 - 54
Mark-Relevant-Passage	8.16	8	5.572	0 - 20
Relevant-Doc	5.96	5	5.465	0 - 21
Non-Relevant-Doc	9.04	5	10.179	0 - 45

Narrative of Search for Topic 236

This seems to have been one of the more frustrating topics for our searcher. For brevity, we do not show the complete ranked lists here.

Are current laws of the sea uniform? If not, what are some of the areas of disagreement?

- **Query** sea marine maritime law legal regulation disagreement dispute
- View rank position 1 (FR88720-0111): Judge not relevant.
- View rank position 2 (FR881121-0087): Judge not relevant.
- View rank position 3 (AP880725-0215): Judge not relevant.
- View rank position 4 (WSJ910626-0071): Judge not relevant.
- View rank position 7 (FR881116-0126)
- **Query** sea law marine legal regulation MARAD
- View rank position 2 (FR88927-0018)
- View rank position 23 (FR881027-0027): Judge *relevant.*
- **Query** sea law regulation ship
- View rank position 1 (FR88120-0060): Judge not relevant.
- View rank position 2 (FR881201-0054): Judge not relevant.
- View rank position 25 (FR88916-0076): Judge not relevant.
- **Query** sea law regulation ship sailing marine permit
- View rank position 4 (FR88506-0038): Judge not relevant.
- **Query** maritime law regulation
- View rank position 1 (FR88212-0072)
- View rank position 2 (FR88726-0078)
- View rank position 3 (WSJ910619-0001)
- **Query** international shipping act regulation law
- View rank position 3 (FR88719-0023): Judge not relevant.
- View rank position 10 (FR881121-0030): Judge *relevant.*
- View rank position 20 (FR88712-0019): Judge not relevant.
- View rank position 25 (FR88712-0040): Judge not relevant.
- **Query** marine law regulation
- View rank position 24 (FR88909-0031)
- **Query** maritime shipping international law
- View rank position 4 (FR881102-0046)
- **Query** maritime shipping fishing internation sea law regulation permit
- View rank position 1 (FR88909-0065)

- View rank position 8 (FR88212-0072)
- View rank position 1 (FR88909-0065): Judge not relevant.
- View rank position 2 (FR88511-0163): Judge not relevant.
- View rank position 16 (FR88923-0151)
- **Query** maritime shipping law regulation international permit dispute disagreement
- View rank position 5 (FR881109-0064): Judge not relevant.
- **Query** sea law shipping act
- **Query** maritime law legal
- **Query** shipping law legal
- View rank position 23 (AP900828-0128): *Mark Relevant Passage.*
- **Feedback**
- View rank position 25 (AP900830-0086)
- **Query** maritime law ship coast board cargo naval Lloyds
- View rank position 1 (AP880809-0243): *Mark Relevant Passage.*
- View rank position 2 (AP900709-0216): *Mark Relevant Passage.*
- View rank position 3 (FR881107-0125)
- **Feedback**
- View rank position 2 (AP880824-0253): Judge not relevant.
- View rank position 3 (AP901015-0272): Judge not relevant.
- View rank position 4 (AP900924-0256): Judge not relevant.
- **Exit!**
- **Total Elapsed Time: 28 minutes**
- **Accumulated Relevant Documents: 5**
  Note: According to the qrel file, none of these were actually relevant.

NATURAL LANGUAGE INFORMATION RETRIEVAL: TREC-4 REPORT

Tomek Strzalkowski
GE Corporate Research & Development
P.O. Box 8
Schenectady, NY 12301
tomek@thuban.crd.ge.com

Jose Perez Carballo[1]
Courant Institute of Mathematical Sciences
New York University
715 Broadway, rm. 704
New York, NY 10003
carballo@cs.nyu.edu

ABSTRACT

In this paper we report on the joint GE/NYU natural language information retrieval project as related to the 4th Text Retrieval Conference (TREC-4). The main thrust of this project is to use natural language processing techniques to enhance the effectiveness of full-text document retrieval. During the course of the four TREC conferences, we have built a prototype IR system designed around a statistical full-text indexing and search backbone provided by the NIST's Prise engine. The original Prise has been modified to allow handling of multi-word phrases, differential term weighting schemes, automatic query expansion, index partitioning and rank merging, as well as dealing with complex documents. Natural language processing is used to (1) preprocess the documents in order to extract content-carrying terms, (2) discover inter-term dependencies and build a conceptual hierarchy specific to the database domain, and (3) process user's natural language requests into effective search queries. The overall architecture of the system is essentially the same as in TREC-3, as our efforts this year were directed at optimizing the performance of all components. A notable exception is the new massive query expansion module used in routing experiments, which replaces a prototype extension used in the TREC-3 system. On the other hand, it has to be noted that the character and the level of difficulty of TREC queries has changed quite significantly since the last year evaluation. TREC-4 new ad-hoc queries are far shorter, less focused, and they have a flavor of information requests (*What is the prognosis of ...*) rather than search directives typical for earlier TRECs (*The relevant document will contain ...*). This makes building of good search queries a more sensitive task than before. We thus decided to introduce only minimum number of changes to our indexing and search processes, and even roll back some of the TREC-3 extensions which dealt with longer and somewhat redundant queries (e.g., locality matching[2]). Overall, our system performed quite well as our position with respect to the best systems improved steadily since the beginning of TREC. It should be noted that the most significant gain in performance seems to occur in precision near the top of the ranking, at 5, 10, 15 and 20 documents. Indeed, our unofficial manual runs performed after TREC-4 conference show superior results in these categories, topping by a large margin the best manual scores by any system in the official evaluation.

INTRODUCTION

A typical (full-text) information retrieval (IR) task is to select documents from a database in response to a user's query, and rank these documents according to relevance. This has been usually accomplished using statistical methods (often coupled with manual encoding) that (a) select terms (words, phrases, and other units) from documents that are deemed to best represent their content, and (b) create an inverted index

[2] This turned out to be a mistake, as we explain later in this paper.

[1] Current address: School of Communication, Information and Library Studies, Rutgers University, New Brunswick, NJ 08903

file (or files) that provide an easy access to documents containing these terms. A subsequent search process will attempt to match preprocessed user queries against term-based representations of documents in each case determining a degree of relevance between the two which depends upon the number and types of matching terms. Although many sophisticated search and matching methods are available, the crucial problem remains to be that of an adequate representation of content for both the documents and the queries.

In term-based representation, a document (as well as a query) is transformed into a collection of weighted terms, derived directly from the document text or indirectly through thesauri or domain maps. The representation is anchored on these terms, and thus their careful selection is critical. Since each unique term can be thought to add a new dimensionality to the representation, it is equally critical to weigh them properly against one another so that the document is placed at the correct position in the N-dimensional term space. Our goal here is to have the documents on the same topic placed close together, while those on different topics placed sufficiently apart. Unfortunately, we often do not know how to compute terms weights. The statistical weighting formulas, based on terms distribution within the database, such as *tf.idf*, are far from optimal, and the assumptions of term independence which are routinely made are false in most cases. This situation is even worse when single-word terms are intermixed with phrasal terms and the term independence becomes harder to justify.

The simplest word-based representations of content, while relatively better understood, are usually inadequate since single words are rarely specific enough for accurate discrimination, and their grouping is often accidental. A better method is to identify groups of words that create meaningful *phrases*, especially if these phrases denote important concepts in the database domain. For example, *joint venture* is an important term in the Wall Street Journal (WSJ henceforth) database, while neither *joint* nor *venture* is important by itself. In the retrieval experiments with the training TREC database, we noticed that both *joint* and *venture* were dropped from the list of terms by the system because their idf (*inverted document frequency*) weights were too low. In large databases, such as TIPSTER, the use of phrasal terms is not just desirable, it becomes necessary.

An accurate syntactic analysis is an essential prerequisite for selection of phrasal terms. Various statistical methods, e.g., based on word co-occurrences and mutual information, are prone to high error rates (sometimes as high as 50%), turning out many unwanted associations. Similarly, simplistic lexical-level parsing methods have limited potential, for example, identifying noun phrases as sequences of adjectives and nouns, adds little value to the document representation beyond what is already provided by the part-of-speech tagging. Further gains are possible if syntactic and semantic level dependencies are identified and represented. Therefore a good, fast parser may be necessary, as we have demonstrated in previous TRECs.

The challenge is to obtain ''semantic'' phrases, or ''concepts'', which would capture underlying semantic uniformity across various surface forms of expression. Syntactic structures are often reasonable indicators of content, certainly better than 'statistical phrases' — where words are grouped solely on the basis of physical proximity (e.g., "college junior" is not the same as "junior college") — however, the creation of compound terms makes the term matching process more complex since in addition to the usual problems of lexical meaning, one must deal with structure (e.g., "college junior" is the same as "junior in college"). In order to deal with structure, the parser's output needs to be "normalized" or "regularized" so that complex terms with the same or closely related meanings would indeed receive matching representations. One way to regularize syntactic structures is to transform them into operator-argument form, or at least head-modifier form, as will be further explained in this paper. In effect, therefore, we aim at obtaining a semantic representation. This result has been achieved to a certain extent in our work thus far.

Do we need to parse indeed? Our recent results indicate that some of the critical *semantic* dependencies can in fact be obtained without the intermediate step of syntactic analysis, and directly from lexical-level representation of text. We have applied our noun phrase disambiguation method directly to word sequences generated using part-of-speech information, and the results were most promising. At this time we have no data how these results compare to those obtained via parsing.

No matter how we eventually arrive at the compound terms, we hope they would let us to capture more accurately the semantic content of a document. It is certainly true that the compound terms such as *South Africa*, or *advanced document processing*, when found in a document, give us a better idea about the content of such document than isolated word matches. What happens, however, if we do not find them in a document? This situation may arise for several reasons: (1) the term/concept is not there, (2) the concept is there but our system is unable to identify it, or (3) the concept is not explicitly there, but its presence can be infered using general or domain-specific knowledge.

This is certainly a serious problem, since we now attach more weight to concept matching than isolated word matching, and missing a concept can reflect more dramatically on system's recall. The inverse is also true: finding a concept where it really isn't makes an irrelevant document more likely to be highly ranked than with single-word based representation. Thus, while the rewards maybe greater, the risks are increasing as well.

One way to deal with this problem is to allow the system to fall back on partial matches and single word matches when concepts are not available, and to use query expansion techniques to supply missing terms. Unfortunately, thesaurus-based query expansion is usually quite uneffective, unless the subject domain is sufficiently narrow and the thesaurus sufficiently domain-specific. For example, the term *natural language* may be considered to subsume a term denoting a specific human language, e.g., *English*. Therefore, a query containing the former may be expected to retrieve documents containing the latter. The same can be said about *language* and *English*, unless *language* is in fact a part of the compound term *programming language* in which case the association *language - Fortran* is appropriate. This is a problem because (a) it is a standard practice to include both simple and compound terms in document representation, and (b) term associations have thus far been computed primarily at word level (including fixed phrases) and therefore care must be taken when such associations are used in term matching. This may prove particularly troublesome for systems that attempt term clustering in order to create "meta-terms" to be used in document representation.

In the remainder of this paper we discuss particulars of the present system and some of the observations made while processing TREC-4 data. While this description is meant to be self-contained, the reader may want to refer to previous TREC papers by this group for more information about the system.

OVERALL DESIGN

Our information retrieval system consists of a traditional statistical backbone (NIST's PRISE system; Harman and Candela, 1989) augmented with various natural language processing components that assist the system in database processing (stemming, indexing, word and phrase clustering, selectional restrictions), and translate a user's information request into an effective query. This design is a careful compromise between purely statistical non-linguistic approaches and those requiring rather accomplished (and expensive) semantic analysis of data, often referred to as 'conceptual retrieval'.

In our system the database text is first processed with a fast syntactic parser. Subsequently certain types of phrases are extracted from the parse trees and used as compound indexing terms in addition to single-word terms. The extracted phrases are statistically analyzed as syntactic contexts in order to discover a variety of similarity links between smaller subphrases and words occurring in them. A further filtering process maps these similarity links onto semantic relations (generalization, specialization, synonymy, etc.) after which they are used to transform a user's request into a search query.

The user's natural language request is also parsed, and all indexing terms occurring in it are identified. Certain highly ambiguous, usually single-word terms may be dropped, provided that they also occur as elements in some compound terms. For example, "natural" is deleted from a query already containing "natural language" because "natural" occurs in many unrelated contexts: "natural number", "natural logarithm", "natural approach", etc. At the same time, other terms may be added, namely those which are linked to some query term through admissible similarity relations. For example, "unlawful activity" is added to a query (TREC topic 055) containing the compound term "illegal activity" via a synonymy link between "illegal" and "unlawful". After the final query is constructed, the database search follows, and a ranked list of documents is returned. In TREC-4, the automatic query expansion has been limited to to routing runs, where we refined our version of massive expansion using relevenace information wrt. the training database. Query expansion via automatically generated domain map was not usd in offical ad-hoc runs.

As in TREC-3, we used a randomized index splitting mechanism which creates not one but several balanced sub-indexes. These sub-indexes can be searched independently and the results can be merged meaningfully into a single ranking.

Before we proceed to discuss the particulars of our system we would like to note that all the processing steps, those performed by the backbone system, and those performed by the natural language processing components, are fully automated, and no human intervention or manual encoding is required.

FAST PARSING WITH TTP PARSER

TTP (Tagged Text Parser) is based on the Linguistic String Grammar developed by Sager (1981). The parser currently encompasses some 400 grammar productions, but it is by no means complete. The parser's output is a regularized parse tree representation of each sentence, that is, a representation that reflects the sentence's logical predicate-argument structure. For example, logical subject and logical object are identified in both passive and active sentences, and noun phrases are organized around their head elements. The parser is equipped with a powerful skip-and-fit recovery mechanism that allows it to operate effectively in the face of ill-formed input or under a severe time pressure. When parsing the TREC-3 collection of more than 500 million words, we found that the parser's speed averaged between 0.17 and 0.26 seconds per sentence, or up to 80 words per second, on a Sun's SparcStation10. In addition, TTP has been shown to produce parse structures which are no worse than those generated by full-scale linguistic parsers when compared to hand-coded Treebank parse trees.

TTP is a full grammar parser, and initially, it attempts to generate a complete analysis for each sentence. However, unlike an ordinary parser, it has a built-in timer which regulates the amount of time allowed for parsing any one sentence. If a parse is not returned before the allotted time elapses, the parser enters the skip-and-fit mode in which it will try to "fit" the parse. While in the skip-and-fit mode, the parser will attempt to forcibly reduce incomplete constituents, possibly skipping portions of input in order to restart processing at a next unattempted constituent. In other words, the parser will favor reduction to backtracking while in the skip-and-fit mode. The result of this strategy is an approximate parse, partially fitted using top-down predictions. The fragments skipped in the first pass are not thrown out, instead they are analyzed by a simple phrasal parser that looks for noun phrases and relative clauses and then attaches the recovered material to the main parse structure. Full details of TTP parser have been described in the TREC-1 report (Strzalkowski, 1993a), as well as in other works (Strzalkowski, 1992; Strzalkowski & Scheyen, 1993).

As may be expected, the skip-and-fit strategy will only be effective if the input skipping can be performed with a degree of determinism. This means that most of the lexical level ambiguity must be removed from the input text, prior to parsing. We achieve this using a stochastic parts of speech tagger to preprocess the text (see TREC-1 report for details).

WORD SUFFIX TRIMMER

Word stemming has been an effective way of improving document recall since it reduces words to their common morphological root, thus allowing more successful matches. On the other hand, stemming tends to decrease retrieval precision, if care is not taken to prevent situations where otherwise unrelated words are reduced to the same stem. In our system we replaced a traditional morphological stemmer with a conservative dictionary-assisted suffix trimmer. [3] The suffix trimmer performs essentially two tasks: (1) it reduces inflected word forms to their root forms as specified in the dictionary, and (2) it converts nominalized verb forms (e.g., "implementation", "storage") to the root forms of corresponding verbs (i.e., "implement", "store"). This is accomplished by removing a standard suffix, e.g., "stor+age", replacing it with a standard root ending ("+e"), and checking the newly created word against the dictionary, i.e., we check whether the new root ("store") is indeed a legal word. Below is a small example of text before and after stemming.

> While serving in South Vietnam, a number of U.S. Soldiers were reported as having been exposed to the defoliant Agent Orange. The issue is veterans entitlement, or the awarding of monetary compensation and/or medical assistance for physical damages caused by Agent Orange.

> serve south vietnam number u.s. soldier expose defoliant agent orange veteran entitle award monetary compensate medical assist physical damage agent orange

Please note that proper names, such as South Vietnam and Agent Orange are identified separately through the name extraction process described below. Note also that various ''stopwords'' (e.g., prepositions, conjunctions, articles, etc.) are removed from text.

HEAD-MODIFIER STRUCTURES

Syntactic phrases extracted from TTP parse trees are head-modifier pairs. The head in such a pair is a central element of a phrase (main verb, main noun, etc.), while the modifier is one of the adjunct arguments of the head. In the TREC experiments reported here we extracted head-modifier word and fixed-phrase pairs only. While TREC databases are large enough to warrant generation of larger compounds, we were unable to verify their effectiveness in indexing, mostly because of the tight schedule.

Let us consider a specific example from the WSJ database:

> The former Soviet president has been a local hero ever since a Russian tank invaded Wisconsin.

The tagged sentence is given below, followed by the regularized parse structure generated by TTP, given in Figure 1.

[3] Dealing with prefixes is a more complicated matter, since they may have quite strong effect upon the meaning of the resulting term, e.g., *un-* usually introduces explicit negation.

> The/*dt* former/*jj* Soviet/*jj* president/*nn* has/*vbz* been/*vbn* a/*dt* local/*jj* hero/*nn* ever/*rb* since/*in* a/*dt* Russian/*jj* tank/*nn* invaded/*vbd* Wisconsin/*np* ./*per*

It should be noted that the parser's output is a predicate-argument structure centered around main elements of various phrases. In Figure 1, BE is the main predicate (modified by HAVE) with 2 arguments (*subject, object*) and 2 adjuncts (*adv, sub_ord*). INVADE is the predicate in the subordinate clause with 2 arguments (*subject, object*). The subject of BE is a noun phrase with PRESIDENT as the head element, two modifiers (FORMER, SOVIET) and a determiner (THE). From this structure, we extract head-modifier pairs that become candidates for compound terms. The following types of pairs are considered: (1) a head noun and its left adjective or noun adjunct, (2) a head noun and the head of its right adjunct, (3) the main verb of a clause and the head of its object phrase, and (4) the head of the subject phrase and the main verb. These types of pairs account for most of the syntactic variants for relating two words (or simple phrases) into pairs carrying compatible semantic content. For example, the pair *retrieve+information* will be extracted from any of the following fragments: *information retrieval system; retrieval of information from databases*; and *information that can be retrieved by a user-controlled interactive search process*. In the example at hand, the following head-modifier pairs are extracted (pairs containing low-content elements, such as BE and FORMER, or names, such as WISCONSIN, will be later discarded):

```
[assert
  [[perf [HAVE]]
    [[verb [BE]]
      [subject
        [np
          [n PRESIDENT]
          [t_pos THE]
          [adj [FORMER]]
          [adj [SOVIET]]]]
      [object
        [np
          [n HERO]
          [t_pos A]
          [adj [LOCAL]]]]
      [adv EVER]
      [sub_ord
        [SINCE
          [[verb [INVADE]]
            [subject
              [np
                [n TANK]
                [t_pos A]
                [adj [RUSSIAN]]]]
            [object
              [np
                [name [WISCONSIN]]]]]]]]]
```

Figure 1. Predicate-argument parse structure.

PRESIDENT+BE, PRESIDENT+FORMER, PRESIDENT+SOVIET, BE+HERO, HERO+LOCAL,
TANK+INVADE, TANK+RUSSIAN, INVADE+WISCONSIN

We may note that the three-word phrase *former Soviet president* has been broken into two pairs *former president* and *Soviet president*, both of which denote things that are potentially quite different from what the original phrase refers to, and this fact may have potentially negative effect on retrieval precision. This is one place where a longer phrase appears more appropriate. The representation of this sentence may therefore contain the following terms (along with their inverted document frequency weights):

PRESIDENT	2.623519
SOVIET	5.416102
PRESIDENT+SOVIET	11.556747
PRESIDENT+FORMER	14.594883
HERO	7.896426
HERO+LOCAL	14.314775
INVADE	8.435012
TANK	6.848128
TANK+INVADE	17.402237
TANK+RUSSIAN	16.030809
RUSSIAN	7.383342
WISCONSIN	7.785689

While generating compound terms we took care to identify 'negative' terms, that is, those whose denotations have been explicitly excluded by negation. Even though matching of negative terms was not used in retrieval (nor did we use negative weights), we could easily prevent matching a negative term in a query against its positive counterpart in the database by removing known negative terms from queries. As an example consider the following fragment from topic 192:

> References to the cost of cleanup and number of people and equipment involved without mentioning the method are not relevant.

The corresponding compound terms are:

> NOT cost cleanup
> NOT number equip
> NOT number people

Note that while this statement is negated, the negation is conditioned with the *without mentioning* ... phrase. Our NLP module is not able to represent such fine distinctions at this time.

NOMINAL COMPOUNDS

The notorious ambiguity of nominal compounds remains a serious difficulty in obtaining head-modifier pairs of highest accuracy. In order to cope with this, the pair extractor looks at the distribution statistics of the compound terms to decide whether the association between any two words (nouns and adjectives) in a noun phrase is both syntactically valid and semantically significant. For example, we may accept *language+natural* and *processing+language* from *natural language processing* as correct, however, *case+trading* would make a mediocre term when extracted from *insider trading case*. On the other hand, it is important to extract *trading+insider* to be able to match documents containing phrases *insider trading sanctions act* or *insider trading activity*. Phrasal terms are extracted in two phases. In the first phase, only unambiguous head-modifier pairs are generated, while all structurally ambiguous noun phrases are passed to the second phase "as is". In the second phase, the distributional statistics gathered in the first phase are used to predict the strength of alternative modifier-modified links within ambiguous phrases. For example, we may have multiple unambiguous occurrences of *insider trading*, while very few of *trading case*. At the same time, there are numerous phrases such as *insider trading case, insider trading legislation*, etc., where the pair *insider trading* remains stable while the other elements get changed, and significantly fewer cases where, say, *trading case* is constant and the other words change.

The disambiguation procedure is performed after the first phrase extraction pass in which all unambiguous pairs (noun+noun and noun+adjective) and all ambiguous noun phrases are extracted. Any nominal string consisting of three or more words of which at least two are nouns is deemed structurally ambiguous. In the Tipster corpus, about 80% of all ambiguous nominals were of length 3 (usually 2 nouns and an adjective), 19% were of length 4, and only 1% were of length 5 or more. The algorithm proceeds in three steps, as follows:

(1) *Assign scores* to each of the candidate pairs x_i+x_j where $i>j$ from the ambiguous noun phrase $x_1 \cdots x_n$. The score assigned to a candidate pair is the sum of the scores for each occurrence of this pair in any compound nominal within the training corpus. For each occurrence, the score is maximum when the words x_i and x_j are the only words in the phrase, i.e., we have unambiguous nominal $x_j x_i$, in which case the score is 1. For longer phrases, for non-adjacent words, and for pairs anchored at words toward the left of the compound, the score decreases proportionately.

(2) For each set $X_j=\{x_i+x_j \mid \text{for } i>j\}$ of candidate pairs *rank* alternative pairs by their scores.

(3) *Disambiguate* by selecting the top choice from each set such that its score is above an empirically established global threshold, it is significantly higher than the second best choice from the set, and it is not significantly lower than the scores of pairs selected from other sets X_i.

The effectiveness of this algorithm can be measured in terms of recall (the proportion of all valid head+modifier pairs extracted from ambiguous nominals), and precision (the proportion of valid pairs among those extracted). The evaluation was done on a small sample of randomly selected phrases, and the algorithm performance was compared to manually selected correct pairs. The following numbers were recorded: recall 66% to 71%; precision 88% to 91%, depending on the size of the training sample. In terms of the total number of pairs extracted unambiguously from the parsed text (i.e., those obtained by the procedure described in the previous section), the disambiguation step recovers an additional 10% to 15% of pairs, all of which were previously thrown out as unrecoverable. A sample set of ambiguous phrases and extracted head+modifier pairs is shown in Table 1.

Ambiguous nominal	Extracted pairs
oil import fee	oil import import fee
croatian wartime cabinet	croatian cabinet wartime cabinet
national enviromental watchdog group	national group enviromental group watchdog group
current export subsidy program	current program export subsidy subsidy program
gas operating and maintaining expenses	**gas operating operating expenses maintaining expenses

Table 1. Ambiguous nominals and extracted pairs.

EXTRACTING PROPER NAMES

Proper names, of people, places, events, organizations, etc., are often critical in deciding relevance of a document. Since names are traditionally capitalized in English text, spotting them is relatively easy, most of the time. Many names are composed of more than a single word, in which case all words that make up the name are capitalized, except for prepositions and such, e.g., *The United States of America.* It is important that all names recognized in text, including those made up of multiple words, e.g., *South Africa* or *Social Security*, are represented as tokens, and not broken into single words, e.g., *South* and *Africa*, which may turn out to be different names altogether by themselves. On the other hand, we need to make sure that variants of the same name are indeed recognized as such, e.g., *U.S. President Bill Clinton* and *President Clinton*, with a degree of confidence. One simple method, which we use in our system, is to represent a compound name dually, as a compound token and as a set of single-word terms. This way, if a corresponding full name variant cannot be found in a document, its component words matches can still add to the document score. A more accurate, but arguably more expensive method would be to use a substring comparison procedure to recognize variants before matching.

In our system names are identified by the parser, and then represented as strings, e.g., *south+africa*. The name recognition procedure is extremely simple, in fact little more than the scanning of successive words labeled as proper names by the tagger (*np* and *nps* tags). Single-word names are processed just like ordinary words, except for the stemming which is not applied to them. We also made no effort to assign names to categories, e.g., people, companies, places, etc., a classification which is useful for certain types of queries (e.g., *To be relevant a document must identify a specific generic drug company*). A more advanced recognizer is planned for TREC-4 evaluation. In the TREC-3 database, compound names make up about 8% of all terms generated. A small sample of compound names extracted is listed below:

right+wing+christian+fundamentalism
u.s+constitution
gun+control+legislation
national+railroad+transportation+corporation
superfund+hazardous+waste+cleanup+programme
u.s+government
united+states
exxon+valdez
dow_corning+corporation
chairman+julius+d+winer
new+york
wall+street+journal
mcdonnell+douglas+corp+brad+beaver
soviet+georgia
rebel+leader+savimbi
plo+leader+arafat
suzuki+samurai+soft_top+4wd
honda+civic
richard+j+rosebery
mr+rosebery
international+business+machine+corp
cytomegalovirus+retinitis
ids+financial+service+analyst+g+michael+kennedy
senate+judiciary+committee
first+fidelity+bank+n.a+south+jersey
eastern+u.s
federal+national+mortgage+association
canadian+airline+international

CREATING AN INDEX

The limited amount of resources that we had available for indexing forced us to devise a method that splits the collection randomly and produces several sub-indexes. This method would allow us now to index even larger collections in reasonable times. The preliminary tests that we carried out in order to compare the performance of systems where the collection is split into N sub-indexes, for different values of N, suggest that a collection can be split into at least 7 sub-indexes without seeing any degradation in the performance. Given the results that we obtained from such tests as well as the fact that the tests were carried out using relatively small collections (about 150 Megabytes) we intend to perform more extensive testing as soon as possible.

One of the problems we had to face for TREC-1 and TREC-2 was that we did not have enough real memory to index the complete collection (category A) in a reasonable time . Even indexing only the collection for category B (550 megabytes for the ad-hoc experiments) used to take 2 weeks, or about 330 hours. This was more slow than the times that could be obtained by other versions of the PRISE system that were already available by that time. We used a slower version because we did not have then enough main memory to use the faster one. The faster version grows the word frequency tree in main memory, and it is the physical memory that matters here, not the virtual memory, since a tree larger than the size of the real memory causes so many page faults that performance becomes unacceptably slow.

The version of the PRISE system that we used for TREC-3 and TREC-4 is much faster than previous versions. According to the on-line documentation provided by NIST the old system would take about 67 hours to index 276 Megabytes of WSJ material while the new system takes less than 2 hours to index the same material. Still, we did not have enough main memory to use the new system to index the complete collection. Our solution to this problem was to split the collection into N sets of almost equal number of documents and create a separate sub-index for each set. In order to keep the N sub-indexes balanced with respect to each other (so that the term idfs are comparable across sub-indexes, for example) we split the collection randomly into N sets. This is done by assigning each document to one of the N sets selected at random. Our goal was to build N sets that would be as homogeneous as possible. At retrieval time the same query is submitted to each one of the sub-indexes and a separate list of ranked documents is obtained for each index. Since we expect idfs to be comparable across sub-indexes, it makes sense to compare the scores of documents belonging to different sub-indexes. The result of the query is then the set of documents with the highest scores chosen from the union of all lists of ranked documents.

In order to evaluate this technique we ran a series of experiments involving about 50000 records. We split that collection into N sets for several values of N (from 1 to 7) and made some measurements of parameters that we expected to be indicators of the degree of homogeneity (e.g., standard deviation of the total number of terms per index, standard deviation of the maximum idf, standard deviation of the number of unique terms, and others). As expected, these indicators showed a decreasing level of homogeneity as N grows larger. This information is summarized in Table 3.

For each value of N, we evaluated the performance of the system using a series of queries for which NIST had provided relevance judgments. For the weighting scheme we were using, and the small collection used for these preliminary experiments, we observed that the performance actually peaks at N = 4 (the average precision when N was 4 was about 7% better than when N was 1). We thought that these results were promising enough to justify the use of the technique described in order to index the complete collection but we intend to perform a much more careful and complete series of experiments as soon as the time and the resources are available. Table 4 summarizes the system's performance at various levels of index split with a subset of AP subcollection.

For TREC-4 we used 7 sub-indexes for the ad-hoc experiments (2200 Megabytes) and 5 for the routing part (1600 Megabytes). We chose these numbers because, in each case, it was the smallest number of sub-indexes that we could handle given our resources. A nice side-effect of this technique is that each index can be created in parallel on a different machine, making the total time required even shorter. The parameters of the 7-way split used in indexing the TREC-4 ad-hoc database are listed in Table 5. The reader may notice that the split is not particularly well balanced, which may be contrasted with a uniform 4-way split used in TREC-3 (cf. TREC-3 proceedings). This may have contributed to a somewhat weaker performance this year.

TERM WEIGHTING ISSUES

Finding a proper term weighting scheme is critical in term-based retrieval since the rank of a document is determined by the weights of the terms it shares with the query. One popular term weighting scheme, known as tf.idf, weights terms proportionately to their inverted document frequency scores and to their in-document frequencies (tf). The in-document

No. of indexes	Max-Mem MB	Max-idf %std	Uniq.terms Mean	Uniq.terms %std
1	81.9	0.000	921253	0.000
2	61.2	0.424	600869	1.006
3	54.7	0.902	438992	11.678
4	48.2	0.555	373249	3.095
5	46.0	0.986	314986	6.356
6	44.1	1.080	279261	7.318
7	46.8	2.432	247606	16.475

Table 3. Statistics of index splitting performed on a subset of Tipster AP88 subcollection consisting of 48,770 records (about 230 MBytes).

No. of indexes	Avg Prec %change	R-Prec %change	Recall %change
1	0.00	0.00	0.00
2	+4.04	+1.85	+1.11
3	+4.63	+0.72	+0.81
4	+7.04	+4.53	+2.59
5	+1.68	+4.08	+3.92
6	+5.68	+2.75	+4.29
7	+4.18	+4.45	+4.36

Table 4. Performance statistics for split index performed on a subset of Tipster AP88 subcollection consisting of 48,770 records (about 230 MBytes).

Index No.	Postings MB	Dict. MB	Max-idf	Records	Uniq.terms
1	60.05	31.08	17.256	78296	1426574
2	55.25	26.57	17.262	78601	1234205
3	57.69	31.10	17.206	75600	1425545
4	66.64	35.34	17.312	81400	1603649
5	60.16	30.40	17.324	82044	1394983
6	59.18	27.83	17.354	83807	1282712
7	65.47	33.33	17.419	87652	1529695

Table 5. Statistics of the 7-way split index created for ad-hoc database from Tipster Disks 2 and 3 (about 2 GBytes).

frequency factor is usually normalized by the document length, that is, it is more significant for a term to occur 5 times in a short 20-word document, than to occur 10 times in a 1000-word article.[4]

In our official TREC runs we used the normalized tf.idf weights for all terms alike: single 'ordinary-word' terms, proper names, as well as phrasal terms consisting of 2 or more words. Whenever phrases were included in the term set of a document, the length of this document was increased accordingly. This had the effect of decreasing tf factors for 'regular' single word terms.

A standard tf.idf weighting scheme (and we suspect any other uniform scheme based on frequencies) is inappropriate for mixed term sets (ordinary concepts, proper names, phrases) because:

(1) It favors terms that occur fairly frequently in a document, which supports only general-type queries (e.g., "all you know about 'star wars'"). Such queries are not typical in TREC.

(2) It attaches low weights to infrequent, highly specific terms, such as names and phrases, whose only occurrences in a document often decide of relevance. Note that such terms cannot be reliably distinguished using their distribution in the database as the sole factor, and therefore syntactic and lexical information is required.

(3) It does not address the problem of inter-term dependencies arising when phrasal terms and their component single-word terms are all included in a document representation, i.e., *launch+satellite* and *satellite* are not independent, and it is unclear whether they should be counted as two terms.

In our post-TREC-2 experiments we considered (1) and (2) only. We changed the weighting scheme so that the phrases (but not the names which we did not distinguish in TREC-2) were more heavily weighted by their idf scores while the in-document frequency scores were replaced by logarithms multiplied by sufficiently large constants. In addition, the top N highest-idf matching terms (simple or compound) were counted more toward the document score than the remaining terms. This 'hot-spot' retrieval option is discussed in the next section.

Schematically, these new weights for phrasal and highly specific terms are obtained using the following formula, while weights for most of the single-word terms remain unchanged:

$$weight(T_i)=(C_1*log(tf)+C_2*\alpha(N,i))*idf$$

In the above, $\alpha(N,i)$ is 1 for $i<N$ and is 0 otherwise. The $\alpha(N,i)$ factor realizes our notion of ''hot spot'' matching, where only top N matches are used in computing the document score. This creates an effect of ''locality'', somewhat similar to that achieved by passage-level retrieval (e.g., Callan, 1994). In TREC-3, where this weighing scheme was fully deployed for the first time, it proved very useful for sharpening the focus of long, frequently convoluted queries. In TREC-3 where the query length ranged from 20 to 100+ valid terms, setting N to 15 or 20 (including phrasal concepts) typically lead to a precision gain of about 20%. In TREC-4, the average query length is less than 10 terms, which we considered too short for using locality matching, and this part of the weighting scheme was in effect unused in the official runs. This turned out to be a mistake, as we rerun TREC-4 experiments after the conference, only to find out that our results improved visibly when the locality part of the weighting scheme was restored.

The table below illustrates the effect of new weighting scheme for phrasal terms using topic 101 (from TREC-2) and a relevant document (WSJ870226-0091).

Topic 101 matches WSJ870226-0091
duplicate terms not shown

TERM	*TF.IDF*	*NEW WEIGHT*
sdi	1750	1750
eris	3175	3175
star	1072	1072
wars	1670	1670
laser	1456	1456

[4] This is not always true, for example when all occurrences of a term are concentrated in a single section or a paragraph rather than spread around the article. See the following section for more discussion.

weapon	1639	1639
missile	872	872
space+base	2641	2105
interceptor	2075	2075
exoatmospheric	1879	3480
system+defense	2846	2219
reentry+vehicle	1879	3480
initiative+defense	1646	2032
system+interceptor	2526	3118
DOC RANK	30	10

Changing the weighting scheme for compound terms, along with other minor improvements (such as expanding the stopword list for topics) has lead to the overall increase of precision of 20% to 25% over our baseline results in TREC-3.

SUMMARY OF RESULTS

The bulk of the text data used in TREC-4 has been previously processed for TREC-3 (about 3.3 GBytes). Routing experiments involved some additional new text (about 500 MBytes), which we processed through our NLP module. The parameters of this process were essentially the same as in TREC-3, and an interested reader is referred to our TREC-3 paper. Two types of retrieval have been done: (1) new topics 201-250 were run in the ad-hoc mode against the Disk-2&3 database,[5] and (2) topics 3-191 (a selection of 50 topics in this range), previously used in TREC-1 to TREC-3, were run in the routing mode against the Disk-1 database plus the new data including material from Federal Register, IR Digest and Internet newsgroups. In each category 2 official runs were performed, with different set up of system's parameters. These runs were labeled *nyuge1* and *nyuge2*, for the routing runs, and *nyuge3* and *nyuge4* for adhoc runs. Both routing runs were automatic, with massive query expansion. Massive query expansion has been implemented as an automatic feedback mode using known relevance judgements for these topics with respect TREC-3 database. The adhoc runs were performed in automatic and manual modes, with nyuge3 being fully automatic, and nyuge4 using manual query expansion before search.

The purpose of the experiments we conducted this year was to find out if some techniques used by other researchers in the past (e.g., massive query expansion) would work well using our NLP techniques. The experiments we tried were the following:

(1) Routing experiment using massive expansion (the official routing run).

(2) Ad-hoc experiment using terms added manually without previous knowledge of the documents (the official ad-hoc manual run).

(3) Ad-hoc experiment using terms selected by a user from documents found in the collection (an un-official ad-hoc semi-interactive run).

Manual ad-hoc experiments. The topics for TREC-4 were much smaller than in previous TREC's. Since our system depends on information obtained by processing the text of the topics, we decided to add text manually. The text added consisted of grammatically correct expressions that we hoped would generate phrases found in relevant documents. The extra text was added without first seeing the documents and relying only on the domain knowledge that the person adding the text might already have. No more than 2 minutes was spend to add text to any query. The results are summarized in Table 6. Notice that the ordering of the experiments with respect to precision is as we expected.

(Semi-)Interactive query expansion ad-hoc experiments. For this experiment we expanded the topics using text taken from the documents. This has been done as follows: a user submited a query and the search was run. The user then reviewed the first two pages of a number of the retrieved documents, and selected phrases from the document's text to be added to the topic. The text added was always full, grammatically correct expressions. The augmented topic were then resubmitted to the system for another process/search cycle. No more than 3 cycles were used. The user spent less than 20 minutess per topic. It should be noted that this expansion did not involve the traditional relevance feedback where terms are added and reweighted based on their distribution in relevant and non-relevant sets (e.g., Roccio formula). Instead, entire phrases and sentences were added, if they appeared to be good extension of the query, which can be considered a natural elaboration of the ''off-the-top-of-your-head'' manual expansion described above. We expect that the same effect could be obtained by expanding the query using a training collection (e.g., Disk 1) different from the retrieval collection, in which case these runs would qualify as manual.

Locality runs. Following the official evaluation, we rerun all adhoc tests using the full scoring scheme that included the locality factor with *N=20*. The results turned out to be visibly better than the official runs, and the summary is given in Table 8. We also compare the locality-enhanced runs with and without phrase matching in Table 9.

Routing experiments. The relevance judgements for the routing queries wrt. the archival data were used to produce a table with 6 columns which contained the

[5] Actually, only 49 topics were used in evaluation, since relevance judgements were unavailable for topic 201 due to an error.

following information:

(1) query number

(2) term taken from the text of the documents

(3) rcount: number of documents that contained the term and were judged relevant.

(4) rtot: total number of documents that were judged relevant.

(5) ncount: number of documents that contained the term and that were judged not relevant.

(6) ntot total number of documents judged not relevant for the corresponding query number.

The weight of each term was computed using the following formula:

$$weight=(rcount/rtot)/(ncount/ntot)$$ [6]

Summary statistics for routing runs are shown in Table 7. All runs shown in this table use massive query expansion.

In general, we can note substantial improvement in performance when phrasal terms are used, especially in ad-hoc runs. Looking back at TREC-2 and TREC-3 one may observe that these improvements appear to be tied to the length and specificity of the query: the longer the query, the more improvement from linguistic processes. This can be seen comparing the improvement over baseline for automatic adhoc runs (very short queries), for manual runs (longer queries), and for semi-interactive runs (yet longer queries). In addition, our TREC-3 results (with long and detailed queries) showed 20-25% improvement in precision attributed to NLP, as compared to 10-16% in TREC-4. At this time we are unable to explain the much smaller improvements in routing evaluations: while the massive query expansion definitely works, NLP has hard time topping these improvements.

CONCLUSIONS

We presented in some detail our natural language information retrieval system consisting of an advanced NLP module and a 'pure' statistical core engine. While many problems remain to be resolved, including the question of adequacy of term-based representation of document content, we attempted to demonstrate that the architecture described here is nonetheless viable. In particular, we demonstrated that natural language processing can now be done on a fairly large scale and that its speed and robustness has improved to the point where it can be applied to real IR problems. We suggest, with some caution until more experiments are run, that natural language processing can be very effective in creating appropriate search queries out of user's initial specifications which can be frequently imprecise or vague. An encouraging thing to note is the sharp increase of precision near the top of the ranking. This indicates a higher than average concentration of relevant documents in the first 10-20 documents retrieved, which can leverage further gains in performance via an automatic feedback process. This should be our focus in TREC-5.

At the same time it is important to keep in mind that the NLP techniques that meet our performance requirements (or at least are believed to be approaching these requirements) are still fairly unsophisticated in their ability to handle natural language text. In particular, advanced processing involving conceptual structuring, logical forms, etc., is still beyond reach, computationally. It may be assumed that these advanced techniques will prove even more effective, since they address the problem of representation-level limits; however the experimental evidence is sparse and necessarily limited to rather small scale tests.

ACKNOWLEDGEMENTS

We would like to thank Donna Harman of NIST for making her PRISE system available to us. Will Rogers provided valuable assistance in installing updated versions of PRISE at NYU. We would also like to thank Ralph Weischedel and Constantine Papageorgiou of BBN for providing and assisting in the use of the part of speech tagger. This paper is based upon work supported by the Advanced Research Projects Agency under Contract N00014-90-J-1851 from the Office of Naval Research, under ARPA's Tipster Phase-2 Contract 94-FI57900-000, and the National Science Foundation under Grant IRI-93-02615.

REFERENCES

Broglio, John and W. Bruce Croft. 1993. "Query Processing for Retrieval from Large Text Bases." Proceedings of ARPA HLT Workshop, March 21-24, Plainsboro, NJ.

Church, Kenneth Ward and Hanks, Patrick. 1990. "Word association norms, mutual information, and lexicography." *Computational Linguistics*, 16(1), MIT Press, pp. 22-29.

Crouch, Carolyn J. 1988. "A cluster-based approach to thesaurus construction." Proceedings of ACM SIGIR-88, pp. 309-320.

[6] Our experiments have shown that this formula may be more effective than the traditional Roccio expansion method (see eg., Frakes & Baeza-Yates, 1992).

Run	abase	nyuge3	mbase	nyuge4	ibase	inlp
Queries	49	49	49	49	49	49
Tot number of docs over all queries						
Ret	46550	46997	49000	48982	49000	49000
Rel	6501	6501	6501	6501	6501	6501
RelRet	2458	2493	3410	3536	3476	3692
%chg		+1.4	+39.0	+44.0	+41.0	+50.0
Recall	Precision					
0.00	0.5296	0.6646	0.7447	0.7377	0.8103	0.8761
0.10	0.3339	0.3733	0.4650	0.5130	0.5423	0.5773
0.20	0.2586	0.2737	0.3724	0.4022	0.4077	0.4464
0.30	0.1939	0.1971	0.2997	0.3304	0.3233	0.3625
0.40	0.1585	0.1641	0.2494	0.2756	0.2740	0.3054
0.50	0.1073	0.1094	0.1714	0.1982	0.2073	0.2391
0.60	0.0831	0.0824	0.1270	0.1363	0.1417	0.1669
0.70	0.0531	0.0505	0.0913	0.0944	0.0968	0.1082
0.80	0.0253	0.0233	0.0509	0.0558	0.0462	0.0499
0.90	0.0058	0.0007	0.0141	0.0201	0.0111	0.0183
1.00	0.0000	0.0000	0.0030	0.0034	0.0006	0.0006
Average precision over all rel docs						
Avg	0.1394	0.1501	0.2082	0.2272	0.2356	0.2605
%chg		+7.7	+49.0	+63.0	+69.0	+87.0
Precision at						
5 docs	0.3755	0.4286	0.5020	0.5469	0.5837	0.6571
10 doc	0.3408	0.3918	0.4510	0.4735	0.5510	0.5898
15 doc	0.3088	0.3619	0.4082	0.4354	0.4857	0.5333
20 doc	0.2857	0.3276	0.3745	0.4163	0.4429	0.4847
30 doc	0.2483	0.2939	0.3503	0.3735	0.4014	0.4333
100 do	0.1624	0.1802	0.2451	0.2545	0.2624	0.2794
200 do	0.1211	0.1315	0.1804	0.1912	0.1869	0.2024
500 do	0.0745	0.0770	0.1069	0.1125	0.1107	0.1189
1000 d	0.0502	0.0509	0.0696	0.0722	0.0709	0.0753
R-Precision (after RelRet)						
Exact	0.1966	0.2088	0.2619	0.2780	0.2834	0.3033
%chg		+6.2	+33.0	+41.0	+44.0	+54.0

Table 6. Ad-hoc runs with queries 202-250: (1) *abase* - automatic run with statistical terms only; (2) *nyuge3* - automatic run with phrases and names; (3) *mbase* - queries manually expanded, but no phrases; (4) *nyuge4* - manual run with phrases; (5) *ibase* - semi-interactive run, no phrases; (6) *inlp* - semi-interactive with phrases.

Run	base	xbase	nyuge1	nyuge2
Queries	50	50	50	50
Tot number of docs over all queries				
Ret	50000	50000	50000	50000
Rel	6576	6576	6576	6576
RelRet	3641	4967	5078	5112
%chg		+36.0	+39.0	+40.0
Recall	(interp) Precision Averages			
0.00	0.5715	0.7420	0.7483	0.7641
0.10	0.3530	0.4898	0.5114	0.5236
0.20	0.2851	0.4220	0.4453	0.4491
0.30	0.2378	0.3614	0.3770	0.3857
0.40	0.1993	0.3145	0.3290	0.3398
0.50	0.1679	0.2730	0.2823	0.2876
0.60	0.1201	0.2285	0.2397	0.2469
0.70	0.0845	0.1701	0.1893	0.1910
0.80	0.0387	0.1263	0.1358	0.1372
0.90	0.0234	0.0652	0.0683	0.0711
1.00	0.0033	0.0067	0.0074	0.0070
Average precision over all rel docs				
Avg	0.1697	0.2715	0.2838	0.2913
%chg		+60.0	+67.0	+72.0
Precision at				
5 docs	0.3760	0.5480	0.5560	0.5680
10 docs	0.3680	0.4840	0.5000	0.5220
15 docs	0.3427	0.4680	0.4880	0.4933
20 docs	0.3240	0.4650	0.4680	0.4800
30 docs	0.3053	0.4447	0.4600	0.4680
100 docs	0.2314	0.3550	0.3658	0.3726
200 docs	0.1791	0.2790	0.2886	0.2931
500 docs	0.1142	0.1655	0.1701	0.1718
1000 docs	0.0728	0.0993	0.1016	0.1022
R-Precision (after Rel)				
Exact	0.2189	0.3100	0.3112	0.3191
%chg		+42.0	+42.0	+46.0

Table 7. Automatic routing runs with 50 queries from 3-191 range: (1) *base* - statistical terms only, no expansion; (2) *xbase* - base run with massive expansion, no phrases; (3) *nyuge1* - syntactic phrases, names, with massive query expansion of up to 500 new terms per query; (4) *nyuge2* - same as 3 but query expansion limited to 200 new terms per query.

Run	abase	aloc	mbase	mloc	ibase	iloc
Queries	49	49	49	49	49	49
Tot number of docs over all queries						
Ret	46550	47013	49000	49000	49000	49000
Rel	6501	6501	6501	6501	6501	6501
RelRet	2458	2498	3410	3545	3476	3723
%chg		+1.6	+39.0	+44.0	+41.0	+51.0
Recall	Precision					
0.00	0.5296	0.6923	0.7447	0.7525	0.8103	0.9071
0.10	0.3339	0.3702	0.4650	0.5326	0.5423	0.6039
0.20	0.2586	0.2821	0.3724	0.4138	0.4077	0.4706
0.30	0.1939	0.2207	0.2997	0.3487	0.3233	0.3936
0.40	0.1585	0.1742	0.2494	0.2941	0.2740	0.3268
0.50	0.1073	0.1345	0.1714	0.2239	0.2073	0.2578
0.60	0.0831	0.0918	0.1270	0.1513	0.1417	0.1836
0.70	0.0531	0.0565	0.0913	0.1027	0.0968	0.1178
0.80	0.0253	0.0295	0.0509	0.0641	0.0462	0.0577
0.90	0.0058	0.0006	0.0141	0.0292	0.0111	0.0227
1.00	0.0000	0.0000	0.0030	0.0045	0.0006	0.0020
Average precision over all rel docs						
Avg	0.1394	0.1592	0.2082	0.2424	0.2356	0.2767
%chg		+14.0	+49.0	+74.0	+69.0	+98.0
Precision at						
5 docs	0.3755	0.4571	0.5020	0.5592	0.5837	0.6694
10 doc	0.3408	0.3939	0.4510	0.4816	0.5510	0.6082
15 doc	0.3088	0.3687	0.4082	0.4490	0.4857	0.5633
20 doc	0.2857	0.3378	0.3745	0.4286	0.4429	0.5133
30 doc	0.2483	0.3075	0.3503	0.3925	0.4014	0.4537
100 do	0.1624	0.1927	0.2451	0.2720	0.2624	0.2978
200 do	0.1211	0.1394	0.1804	0.2051	0.1869	0.2124
500 do	0.0745	0.0798	0.1069	0.1176	0.1107	0.1240
1000 d	0.0502	0.0510	0.0696	0.0723	0.0709	0.0760
R-Precision (after RelRet)						
Exact	0.1966	0.2211	0.2619	0.2934	0.2834	0.3205
%chg		+12.0	+33.0	+49.0	+44.0	+63.0

Table 8. Ad-hoc runs with queries 202-250: (1) *abase* - automatic run with statistical terms only; (2) *aloc* - automatic run with phrases and names and locality factor set at N=20; (3) *mbase* - run with queries manually expanded, but no phrases; (4) *mloc* - manual run with phrases and locality N=20; (5) *ibase* - semi-interactive run, no phrases; (6) *iloc* - semi-interactive run with phrases, locality N=20.

	no pairs	with pairs	% change
Automatic (no locality)			
avg prec	0.1394	0.1501	7.68
prec at 10	0.3339	0.3733	11.78
Automatic (with locality)			
avg prec	0.1555	0.1592	2.38
prec at 10	0.3434	0.3702	7.80
Manual (no locality)			
avg prec	0.2082	0.2272	9.13
prec at 10	0.4650	0.5130	10.32
Manual (with locality)			
avg prec	0.2252	0.2424	7.64
prec at 10	0.4843	0.5326	9.97
Semi-Interactive (no locality)			
avg prec	0.2372	0.2626	10.71
prec at 10	0.5471	0.5843	6.80
Semi-Interactive (with locality)			
avg prec	0.2533	0.2767	9.24
prec at 10	0.5679	0.6039	6.34

Table 9. Effect of locality weighting in adhoc runs.

Frakes, William, B. and Ricardo Baeza-Yates. (eds). 1992. *Information Retrieval* Prentice-Hall, Englewood Cliffs, NJ.

Grefenstette, Gregory. 1992. ''Use of Syntactic Context To Produce Term Association Lists for Text Retrieval.'' Proceedings of SIGIR-92, Copenhagen, Denmark. pp. 89-97.

Grishman, Ralph, Lynette Hirschman, and Ngo T. Nhan. 1986. ''Discovery procedures for sublanguage selectional patterns: initial experiments''. *Computational Linguistics,* 12(3), pp. 205-215.

Grishman, Ralph and Tomek Strzalkowski. 1991. ''Information Retrieval and Natural Language Processing.'' Position paper at the workshop on Future Directions in Natural Language Processing in Information Retrieval, Chicago.

Harman, Donna. 1988. ''Towards interactive query expansion.'' Proceedings of ACM SIGIR-88, pp. 321-331.

Harman, Donna and Gerald Candela. 1989. ''Retrieving Records from a Gigabyte of text on a Minicomputer Using Statistical Ranking.'' *Journal of the American Society for Information Science,* 41(8), pp. 581-589.

Hindle, Donald. 1990. ''Noun classification from predicate-argument structures.'' Proc. 28 Meeting of the ACL, Pittsburgh, PA, pp. 268-275.

Kwok, K.L., L. Papadopoulos and Kathy Y.Y. Kwan.

1993. ''Retrieval Experiments with a Large Collection using PIRCS.'' Proceedings of TREC-1 conference, NIST special publication 500-207, pp. 153-172.

Lewis, David D. and W. Bruce Croft. 1990. ''Term Clustering of Syntactic Phrases''. Proceedings of ACM SIGIR-90, pp. 385-405.

Meteer, Marie, Richard Schwartz, and Ralph Weischedel. 1991. "Studies in Part of Speech Labeling." Proceedings of the 4th DARPA Speech and Natural Language Workshop, Morgan-Kaufman, San Mateo, CA. pp. 331-336.

Sager, Naomi. 1981. *Natural Language Information Processing.* Addison-Wesley.

Sparck Jones, Karen. 1972. ''Statistical interpretation of term specificity and its application in retrieval.'' *Journal of Documentation,* 28(1), pp. 11-20.

Sparck Jones, K. and E. O. Barber. 1971. ''What makes automatic keyword classification effective?'' *Journal of the American Society for Information Science,* May-June, pp. 166-175.

Sparck Jones, K. and J. I. Tait. 1984. ''Automatic search term variant generation.'' *Journal of Documentation,* 40(1), pp. 50-66.

Strzalkowski, Tomek and Barbara Vauthey. 1991. ''Fast Text Processing for Information Retrieval.'' Proceedings of the 4th DARPA Speech and Natural Language Workshop, Morgan-Kaufman, pp. 346-351.

Strzalkowski, Tomek and Barbara Vauthey. 1992. ''Information Retrieval Using Robust Natural Language Processing.'' Proc. of the 30th ACL Meeting, Newark, DE, June-July. pp. 104-111.

Strzalkowski, Tomek. 1992. ''TTP: A Fast and Robust Parser for Natural Language.'' Proceedings of the 14th International Conference on Computational Linguistics (COLING), Nantes, France, July 1992. pp. 198-204.

Strzalkowski, Tomek. 1993. ''Natural Language Processing in Large-Scale Text Retrieval Tasks.'' Proceedings of the First Text REtrieval Conference (TREC-1), NIST Special Publication 500-207, pp. 173-187.

Strzalkowski, Tomek. 1993. ''Robust Text Processing in Automated Information Retrieval.'' Proc. of ACL-sponsored workshop on Very Large Corpora. Ohio State Univ. Columbus, June 22.

Strzalkowski, Tomek and Jose Perez-Carballo. 1994. ''Recent Developments in Natural Language Text Retrieval.'' Proceedings of the Second Text REtrieval Conference (TREC-2), NIST Special Publication 500-215, pp. 123-136.

Strzalkowski, Tomek, Jose Perez-Carballo and Mihnea Marinescu. 1995. ''Natural Language Information Retirieval: TREC-3 Report.'' Proceedings of the Third Text REtrieval Conference (TREC-3), NIST Special Publication 500-225, pp. 39-53.

Strzalkowski, Tomek. 1995. ''Natural Language Information Retrieval'' **Information Processing and Management**, Vol. 31, No. 3, pp. 397-417. Pergamon/Elsevier.

Strzalkowski, Tomek, and Peter Scheyen. 1993. ''An Evaluation of TTP Parser: a preliminary report.'' Proceedings of International Workshop on Parsing Technologies (IWPT-93), Tilburg, Netherlands and Durbuy, Belgium, August 10-13.

Multi-lingual Text Filtering Using Semantic Modeling

James R. Driscoll*

and

Sara Abbott, Kai-Lin Hu, Michael Miller, Gary Theis

Department of Computer Science
University of Central Florida
Orlando, Florida 32816

Abstract

Semantic Modeling is used to investigate multilingual text filtering. In our approach, the Entity-Relationship (ER) Model is used as a basis for descriptions of information preferences (profiles) in the information filtering process. A profile is viewed as having both a static aspect and a dynamic aspect. The static aspect of a profile can be represented as an ER schema; and the dynamic aspect of the profile can be represented by synonyms of schema components and domain values for schema attributes. For TREC-4, the routing task and the Spanish adhoc task are accomplished using this technique. For the routing task, a large amount of time was spent in an effort to optimize filter performance using the training data that was available for the routing topics. For the Spanish adhoc task, a large amount of time was spent using external sources to develop good filters; in addition, some time was spent implementing a program to help port our approach to this second language. A multi-lingual (English, French, German, and Spanish) experiment is also reported.

Introduction - The Filtering Task and Profiles

Our approach to filtering was first presented at TREC-3 [4]. The filtering process is based on descriptions of individual (or group) information preferences, often called profiles. Profiles typically represent long-term interests. Information filtering is concerned with repeated uses of the profiles, and the profile is assumed to be a correct specification of an information interest [2].

We view a profile as having both a static aspect and a dynamic aspect. We present a procedure for representing the user need statement of a TREC topic as a database Entity-Relationship (ER) schema. An ER schema becomes the static aspect of a profile. For a schema, a synonym list is created for each of the schema components, and a domain value list is created for each of the schema attributes. These lists become the major part of the dynamic aspect of a profile. Our filtering procedure uses the dynamic aspect of a profile to detect relevant documents.

TREC topics are descriptions of information to be considered relevant, and these must be transformed into filter profiles. Each TREC topic is in the form of a highly-formatted, natural language, user need statement. Refer to Figure 1 for an example. This is TREC Topic 173 which concerns smoking bans.

A discussion of semantic modeling and our procedure for making a profile which represents a TREC topic appears in [4]. In this paper, we explain our approach using our TREC-4 filter for Topic 173 as an example.

* Dr. Driscoll's current affiliation is Praxis Technologies, 280 West Canton Avenue, Suite 230, Winter Park, Florida, 32789. Phone: (407) 647-0001, Fax: (407) 628-1832, E-mail: driscoll@prx.com or jdriscol@prx.com.

<top>

<num> Number: 173

<title> Topic: Smoking Bans

<desc> Description:

Document will provide data on smoking bans initiated worldwide in the public and private sector workplace, on various modes of public transportation, and in commercial advertising.

<narr> Narrative:

A relevant document would include data on smoking bans that have been initiated worldwide in the workplace, on various modes of transportation, and in commercial advertising. Relevant information would include such data as who initiated the ban, affected areas, enforcement policy/procedures if any, and any penalties that may be imposed. Also relevant would be tobacco company reactions, legislation imposing a smoking ban, and legal actions or court decisions pertaining to a smoking ban. Not relevant would be documents containing public or private sector comments on smoking in general but not related to a specific ban. Also not relevant would be documents related to health hazards associated with smoking and not referring to a smoking ban.

</top>

Figure 1. TREC Topic 173.

Details and an Example

In our approach, a filter for a TREC topic includes a series of lists (files). A list is either a synonym list or a domain list. Refer to [4] for the theory behind synonym and domain lists. A synonym list is just a list of words or phrases that indicate the same concept. For example, synonyms for the word “ban”. Sometimes, several synonym lists can be merged into one list. We also include the various forms of a word in synonym lists. A domain list is just a list of words or phrases that represent the various allowed values for a particular item. For example, the names of public places could be banks, department stores, elevators, hospitals, etc. Sometimes, the values that should be in a domain list are not known, or only partially known.

Our approach allows weights to be assigned to indicate the importance of each list. We also make use of a window of text when text is examined. Our system has the ability to count "hits" in a window for various combinations of the lists. The window size, the file combinations, and file weights can be adjusted.

At the present time we manually create filter profiles. We determine the significant domain and synonym lists from a study of the TREC topic. Then we initially populate the lists using thesauri, dictionaries, and whatever other reference sources we can find; sometimes a list remains empty. Populating the lists in this manner is the procedure we used to build filters for the Spanish adhoc task. For the routing task, we used training data by reading relevant and not-relevant documents to build filters. Finally, we create an information (INF) file to specify the window size, and the various file combinations and file weights.

To establish a filter for a TREC topic, and do TREC filtering experiments, we have a standard text scanning program which inputs the window size, the domain and synonym files, and one or more variations (which also indicate weights) of the lists. The scanning program then moves across TREC document collections, producing a ranked list of relevant documents. We have used the TREC training data to modify the dynamic aspect of a profile. This is accomplished by using viewed relevant and non-relevant documents to adjust the window size and make additions and deletions to the domain and synonym lists. We have developed a few utility programs to help us do this quickly.

In its current implementation, the scanner requires a stream of text delimited by standard SGML. The only specific markers actually used are the <DOC>, </DOC>, <DOCNO>, and </DOCNO> markers.

Figure 2 provides an example of a filter for TREC Topic 173 which appears in Figure 1. The INF file in Figure 2 indicates the topic number, the size of the text window to be used, the number of synonym and domain files, an output filename, the actual file names of the synonym and domain files, a minimum document relevancy value to consider valid, and the number of combinations followed by the weighted combinations that could indicate a relevant piece of information.

Figure 2 also reveals the synonym and domain files which are specified in the INF file. There are five files specified and they are named to somewhat indicate their content. The filter is looking for the following information:

a. The name of an action.
b. The name of an affected area.
c. A synonym for the word "ban".
d. The name of an indicator for human use of tobacco.
e. A descriptive phrase for a smoking ban.

A central part of the scanner is the "text window". This structure is essentially an array which contains the current group of words being evaluated for local purposes. At any given point in processing, the text window contains the last *X* words read from the text, where *X* is specified as the window size in the INF file. For Topic 173, *X* is 300 as shown in Figure 2. This provides for a variety of local evaluation sizes (e.g., searching on a paragraph-by-paragraph size of roughly 100 words, as opposed to a sentence-by-sentence search of 20 words at a time). The text window's usage gives rise to the terms local and global. Local refers to an evaluation done exclusively on the text within the window, and global refers to an evaluation on the entire text of a document.

Documents are evaluated in a single-pass through the text. Document text begins immediately after the document ID, and ends at the document end marker. As each new word is scanned into the text window, it is compared to the entries in the files. This is accomplished by having read all the file entries into a memory resident hash table prior to the scanning process. If a match or matches are found, they are tallied in an array which contains the number of matches currently within the window, by file. At the same time, match counts that are passing out the end of the text window are subtracted from the array.

When the current word registers a match, there is an immediate evaluation of the current window's contents. For each valid combination specified in the INF file, there is determined a combination value. The combination value is the sum of the quotients of the number of non-zero, required matches (zero matches or non-required files add zero to the sum) for each file multiplied by 1 minus the result of 1 divided by the sum of the "parts" specified in the INF file for each particular file in the combination being evaluated plus the total number of files.

Only the highest combination value encountered is retained, such that at the end of the document, there will be a set of combination values which are maximums for the entire document. Concurrent with these local evaluations, a global document match array is maintained for combined local and global weighting at the end of the document.

Once the entire text of a document has been scanned, a local weight is determined by summing the squares of the best combination weights achieved within the document. Following this, a series of combination values are calculated, then summed, and squared to arrive at a global weight. This operation is identical to the combination calculations for local weight except that the global match count array is used instead of the array for the window. The final weight of the document is then reported as 75% of the local weight added to 25% of the global weight.

The filter shown in Figure 2 was built in two hours by Kai-Lin Hu. Several existing files from the filter for Topic 125 were used. Entries in the few files created for this filter are words and phrases (and their variants) found by reading some of the known relevant documents (the training data) for this topic. This filter achieved best performance for Topic 173.

Topic 173 - Smoking Bans

INF File For Topic 173

```
173
300
5
t173.out 0
t173.action_name.dom 1
t173.affected_area.dom 1
t173.ban.syn 1
t173.human_use.dom 1
t173.smoking_ban.dom 1
0.0
1
5 1 1 5 13
```

t173.action_name.dom
name of an action
(domain file - from Topic 125 filter)

```
CURTAILED ADVERTISING,
NON SMOKING AREA,
NON SMOKING AREAS,
NON-SMOKING AREA,
NON-SMOKING AREAS,
SPECIAL TAX,
SPECIAL TAXES,#
```

t173.afftected_area.dom
name of an affected area
(domain file - specified in Topic 173)

```
BANKS,
DEPARTMENT STORE,
DEPARTMENT STORES,
DOMESTIC FLIGHTS,
ELEVATOR,
ELEVATORS,
HOSPITAL,
HOSPITALS,
HOTEL,
HOTELS,
LARGE STORE,
LARGE STORES,
RESTAURANT,
RESTAURANTS,
TAXICAB,
TAXICABS,
THEATER,
THEATERS,
THEATRE,
THEATRES,
WORKPLACE,
WORKPLACES,#
```

t173.ban.syn
synonym for ban
(synonym file - specified in Topic 173)

```
BAN,
BANNED,
BANNING,
BANS,
FINE,
PENALIZE,
PENALIZED,
PENALIZES,
PENALIZING,
PENALTY,
PROHIBIT,
PROHIBITED,
PROHIBITING,
PROHIBITS,
RESTRICT,
RESTRICTED,
RESTRICTING,
RESTRICTION,
RESTRICTIONS,
RESTRICTS,#
```

t173.human_use.dom
name of indicator for human use of tobacco
(domain file - from Topic 125 Filter)

```
CIGARETTE,
CIGARETTES,
SMOKE,
SMOKED,
SMOKER,
SMOKERS,
SMOKES,
SMOKING,#
```

t173.smoking_ban.dom
descriptive phrase for a smoking ban
(domain file - specified in Topic 173)

```
AGAINST SMOKER,
AGAINST SMOKERS,
AGAINST SMOKING,
ANTI SMOKING,
ANTI-SMOKING,
ATTACK ON SMOKING,
ATTACK SMOKING,
ATTACKED SMOKING,
ATTACKING SMOKING,
ATTACKS ON SMOKING,
ATTACKS SMOKING,
BAN ADVERTISEMENT BY TOBACCO,
```

Figure 2. A Filter for TREC Topic 173 (continued on next page).

BAN ADVERTISEMENTS BY TOBACCO,
BAN CIGARETTE SMOKING,
BAN ON PUBLIC SMOKING,
BAN ON SMOKING,
BAN SALES OF CIGARETTES,
BAN SMOKING,
BAN THE SALE OF KENTS,
BANNED CIGARETTE SMOKING,
BANNED SMOKING,
BANNING CIGARETTE SMOKING,
BANNING SMOKING,
BANNING THE SALE OF KENTS,
BANS ADVERTISEMENT BY TOBACCO,
BANS ADVERTISEMENTS BY TOBACCO,
BANS CIGARETTE SMOKING,
BANS ON PUBLIC SMOKING,
BANS ON SMOKING,
BANS SMOKING,
CIGARETTE BAN,
CIGARETTE BANS,
CIGARETTE-BAN,
CIGARETTE-BANS,
CIGARETTES OUT OF THE WAY,
FORBID SMOKING,
FORBIDDEN SMOKING,
FORBIDDING SMOKING,
FORBIDS SMOKING,
INCREASE TAXES ON TOBACCO,
INCREASED TAXES ON TOBACCO,
INCREASES TAXES ON TOBACCO,
INCREASING TAXES ON TOBACCO,
NO SMOKING RULE,
NO SMOKING RULES,
NO-SMOKING DAY,
NO-SMOKING DAYS,
NO-SMOKING RULE,
NO-SMOKING RULES,
NON SMOKERS EQUAL RIGHT,
NON SMOKERS EQUAL RIGHTS,
NON-SMOKERS EQUAL RIGHT,
NON-SMOKERS EQUAL RIGHTS,
NON-SMOKING DAY,
NON-SMOKING DAYS,
NON-SMOKING MOVEMENT,
NON-SMOKING MOVEMENTS,
NONSMOKING DAY,
NONSMOKING DAYS,
NONSMOKING MOVEMENT,
NONSMOKING MOVEMENTS,
NOT TO SMOKE,
OFF LIMITS TO SMOKERS,
OFF-LIMITS TO SMOKERS,
ONLY FOR NON-SMOKER,
ONLY FOR NON-SMOKERS,
OUTLAW TOBACCO,
PENALIZE SMOKER,
PENALIZE SMOKERS,
PENALIZES SMOKERS,
PENALIZING SMOKERS,
PROHIBIT ADVERTISING OF CIGARETTES,
PROHIBIT SMOKING,
PROHIBIT TEEN-AGER SMOKING,
PROHIBIT TEEN-AGERS FROM SMOKING,
PROHIBIT TEENAGER SMOKING,
PROHIBIT TEENAGERS FROM SMOKING,
PROHIBIT TOBACCO,
PROHIBITED SMOKING,
PROHIBITED TOBACCO,
PROHIBITING SMOKING,
PROHIBITING TOBACCO,
PROHIBITION AGAINST CIGARETTE SMOKING,
PROHIBITION AGAINST SMOKING,
PROHIBITIONS AGAINST CIGARETTE SMOKING,
PROHIBITIONS AGAINST SMOKING,
PROHIBITS ADVERTISING OF CIGARETTES,
PROHIBITS SMOKING,
PROHIBITS TOBACCO,
RAISE TAXES ON TOBACCO,
RAISING TAXES ON TOBACCO,
RESTRICT THE RIGHTS OF SMOKERS,
RESTRICTED THE RIGHTS OF SMOKERS,
RESTRICTING THE RIGHTS OF SMOKERS,
RESTRICTS THE RIGHTS OF SMOKERS,
SMOKE FREE ENVIRONMENT,
SMOKE FREE ENVIRONMENTS,
SMOKE FREE NORWAY,
SMOKE FREE,
SMOKE-FREE,
SMOKELESS,
SMOKING BAN,
SMOKING BANS,
SMOKING HAS BEEN BANNED,
SMOKING IS ALSO PROHIBITED,
SMOKING IS BANNED,
SMOKING IS PROHIBITED,
SMOKING RESTRICTION,
SMOKING RESTRICTIONS,
SMOKING WAS BANNED,
SMOKING WILL BE BANNED,
SMOKING-BAN,
SMOKING-BANS,
STASH THE CIGARETTE,
STASH THE CIGARETTES,
STOP SELLING TOBACCO,
STOP-SMOKING,
STOPPED SELLING TOBACCO,
STOPPING SELLING TOBACCO,
STOPS SELLING TOBACCO,#

Figure 2 (continued from previous page).

Spanish Adhoc Retrieval

Our participation in the Spanish adhoc task originated as Sara Abbott's semester project for an undergraduate course in *Data Processing Systems Implementation.* During the 1995 spring semester, our filtering system was modified to handle the special 8-bit characters found in the Spanish text, and an auxiliary program was written to "expand" Spanish verbs and adjectives listed in synonym and domain files in their infinitive and masculine singular forms, respectively. As a summer semester independent study project, Sara developed filters (synonym, domain, and INF files) for one run on Topics SP1-SP25, and two runs on Topics SP26-SP50. Sara speaks Spanish as a second language. She had previous experience submitting TREC results by participating in the TREC-3 Category B routing task using her own text scanning system; however, she had no previous experience using our present filtering system.

Modification to Accommodate Special 8-bit Characters

In the process of *ftp*-ing and decompressing the approximately 200 megabytes of Spanish text, the 8-bit characters relevant to our task, *á*, *é*, *í*, *ó*, *ú*, *ü*, *ñ*, and *Ñ*, were translated to other extended *ASCII* characters, such as Greek letters and mathematical symbols. Rather than investigate the cause of this undesired translation, we chose to simply translate the symbols to appropriate uppercase unaccented and unumlauted vowels and to restore the letter *Ñ* in its uppercase form. This works because our filtering system converts all of the regular ASCII characters to uppercase before making hash table comparisons.

We treat accented and umlauted vowels as regular vowels because accent marks and umlauts in Spanish generally serve no other purpose than to indicate an irregularly accented syllable, an irregular pronunciation, or a variation in verb form. These diacritical marks are also often inadvertently or deliberately omitted in Spanish text. We found this to be the case, at times, in the *El Norte* collection. It should be noted that this collection contains no uppercase versions of accented or umlauted vowels, possibly due to the unavailability of some of these characters in the keyboard configuration used to create the text. Since our system ignores most punctuation, we ignore the ¿ and ¡ symbols. Only six lines were added to the source code of our filtering system to handle Spanish special characters.

Auxiliary Program for Conjugation of Spanish Verbs

Perhaps the most laborious part of the adaptation of our filter system to accommodate Spanish was the creation of an auxiliary program to generate various Spanish verb forms, when given the infinitive form. This was implemented by allowing placement of the infinitive form, followed by an asterisk flag, in a preliminary synonym or domain list. Given the preliminary list as standard input, a *flex* program, which we call *lexpand*, produces an equivalent longer list containing all useful forms of "flagged" infinitives. Words and phrases which are not flagged remain unaltered. Flagged adjectives ending in *O* are expanded to include feminine and plural forms. Appendix A shows an example of a preliminary synonym list, along with a verb and an adjective from the list as expanded by *lexpand*.

Since *lexpand* at times generates nonsensical verb forms which would never occur (*comermelo*, for example), and includes *vosotros* forms, which are not generally used in Mexico, other than in Biblical references, our philosophy is essentially "better too much than not enough." The junk verbs that we generate only slightly hinder efficiency. The advantage of this approach over stemming is that we can edit the lists generated by *lexpand* to exclude any verb that we do not like. In this manner we avoid some of the following incorrect identifications:

Problem Word	English Meaning	Mistaken For	English Meaning
para	for	*para*	stops
cómo, como	how, like	*como*	eat
nada	nothing	*nada*	swims
vino	wine	*vinó*	came

The *lexpand* program, in its present state, consists of about 700 lines of *flex* source code, which, in turn, generates about 2500 lines of *C* code. It could be improved if broken up into more modules, and through the use of more tables. *Flex* was chosen over other available lexical analyzers because it generates fast executable programs and has fewer restrictions on the size of the tables that it produces, since it dynamically allocates table space.

Construction of Spanish Queries

Spanish queries were constructed manually, and like our English queries (filters) for the routing task, consisted of collections of key phrases and words, referenced by INF files. Although no actual documents were read as we formed TREC-4 Spanish queries, some phrase collections were developed by extracting single lines of text containing key words from the Spanish adhoc document collection and then building lists of phrases with similar meanings. Synonym and domain lists were developed with the help of friends in the local Latin-American community, and with the aid of the following dictionaries and thesauri:

Diccionario de Sinónimos Explicados,
Alonzo, Martín [1].

Larousse Diccionario Escolar,
García-Pelayo y Gross, Ramón [7].

The New World SPANISH-ENGLISH and ENGLISH-SPANISH Dictionary,
Ramondino, Salvatore (editor) [10].

We found an abundance of information related to Mexican commerce and politics, and even some specific names of corporations and trade promotion organizations in the following reference:

Mexico Business: The Portable Encyclopedia for Doing Business with Mexico,
Nolan, James L. etal [9].

Description of Spanish Adhoc Runs

Our two runs for Spanish Topics 26-50, UCFSP1 and UCFSP2, use essentially the same collections of phrases and words for each query. UCFSP1, which was our primary run for official evaluation, is a conservative run, using only single weighting patterns and no negative weighting. UCFSP2 was made using a slight modification to the filter operation described in an earlier section. The modification was to select the highest pattern weight for each window as that window's weight, and to use the highest global weight generated by any pattern. Seventy-five percent of the square of the window weight was then added to twenty-five percent of the square of the global weight to yield a total weight for the given document.

The filter operation used for the other Spanish runs (and the English runs) takes the sum of the squares of all pattern weights for both the window weight and global weight, rather than choosing the highest weights. Squaring the highest pattern weight instead of summing the squares of all pattern weights means that the scanner is now choosing between patterns rather than "averaging" all patterns within a given document.

The filter operation for the UCFSP2 run also allows for lists to receive negative weights within patterns. The assignment of negative weight to a list seeks to increase precision, by subtracting weight when certain words or phrases are encountered. The strategy is that the presence of certain words or phrases can hint that positively weighted words or phrases are being taken out of context. Topic SP41, for example, deals with measures taken in Mexico to control or limit flooding. Since there was extensive flooding along the Mississippi River during 1993, it would be reasonable to expect a query for words equivalent to *flood* and *flooding* to retrieve some documents pertaining to flooding in the United States. A list was constructed for this topic consisting of the names of states along the Mississippi and the list was assigned a negative weight.

Problemas, Problemas, y Más Problemas

As soon as we had our filter system Spanish-ready, we began making filters for the Spanish Topics SP1-SP25 from TREC-3 because qrels were available for these topics. Chris Buckley from Cornell sent us the evaluations from one of their runs to use as a benchmark. Our performance was, generally, considerably lower. However, we found that by eliminating entire domain and synonym lists in a query (filter), we could match or exceed the benchmark. On one topic, a precision of above .8900 was achieved using a single synonym list containing just six words! To us, this meant that the qrels were not very useful.

Even so, we thought that we could use the qrels to at least get some indication of our performance. This was not the case. After reading only a few of the not-relevant documents that we retrieved, and relevant documents that we did not retrieve, it became obvious that running the evaluator using judgments from TREC-3 gave virtually no real indication of performance. We were actually making our queries worse as we tried to approach benchmark precision.

We had expected to retrieve some documents that were not retrieved last year, due to the small number of participants last year. What we didn't expect were the many not-relevant documents that were judged to be relevant in the documents that we examined. Topic SP5 deals with *maquiladoras*, or in-bond manufacturing enterprises. It is understandable how some scanning methods, particularly those using trigrams, might pick up a document containing the word *maquillaje*, meaning "the application of cosmetics," but it was not expected that such a document could be judged to be relevant. Document SP94-0032014, which was judged to be relevant for Topic SP5, is entitled "*Ofrece clases de maquillaje*," or "Cosmetology classes are offered," and goes on to describe the classes offered.

This is perhaps an extreme example of how odd the TREC-3 judgments were. All of the topics that we looked at had many not-relevant documents judged to be relevant. Usually the lack of relevancy was a bit more subtle. Topic SP1, for example, dealt with Mexican opposition to NAFTA. We found that there were more documents dealing with U.S. or Canadian opposition, or that were neutral or pro-NAFTA than there were truly relevant documents in the ones rated as relevant. It appears that if the document contained a reference to NAFTA, it was judged as relevant.

Unfortunately, we did not have time to do a really good set of runs for the Topics SP1-SP25, after the discovery that the qrels were not useful. The run which we submitted, UCFSP0, was made with very few, very short synonym and domain lists, but should still be somewhat useful in the development of new qrels.

Another problem which we encountered with two topics, one from Topics SP1-SP25, and the other from Topics SP26-SP50, was that the event which the topic dealt with occurred after the documents in the collection were written! The *El Norte* collection appears to be dated from December of 1992 through September of 1993. Topic SP2 dealt with the political effects of the assassination of Mexican presidential candidate Luís Donaldo Colosio Murrieta, who was slain on March 23, 1994 [6]. There can be no relevant documents on this topic, although one was found last year. Topic SP31 dealt with measures taken by the Mexican government to resolve a "dispute" in the Mexican state of Chiapas, which actually began on New Year's Day of 1994, when the Zapatista National Liberation Army declared war against the government and seized four towns [6]. There are documents in the collection which mention rebels in Chiapas, and that are related to Topic SP13, which deals with confrontations between the Mexican army and suspected Zapatista rebels, but it is our belief that the "dispute" referred to in the topic description was actually intended to mean the full scale dispute that erupted on January first.

As evidenced above, by the questionable meaning of the word "dispute," the shortened topic descriptions for TREC-4 Spanish topics are somewhat vague, and at times we were unsure what to look for. The description for Topic SP42, as another example, consists of only the question, "Will NAFTA be successful in Mexico?" We didn't know whether to look for documents stating that NAFTA would be a success, or those just stating an opinion one way or the other. We felt that the descriptions for Topics SP1-SP25 were a lot more specific and easier to work with than those for Topics SP26-SP50.

Network Environment

The computer network that enabled undergraduate students to develop filters for TREC-4 is the same one we used for TREC-3 [4].

For training routing filters on just the Vol. 1 and Vol. 2 CDs:

1. The Vol. 1 CD was copied to the hard drive of a PC running Linux (a public domain version of Unix) and functioning as an NFS node on the network.

2. The Vol. 2 CD was copied to the hard drive of a SPARC Server 690MP (four processors) on the network.

3. Students ran filters and viewed training text from 32 RISC machines across the network.

For the UCF100, UCFSP0, UCFSP1, and UCFSP2 runs:

1. The routing and Spanish adhoc document collections were copied to the hard drive of the SPARC Server 690MP (four processors) on the network.

2. Most filters were run on the SPARC Server 690MP. (A few were run on the RISC machines.)

Each of the 32 RISC 6000 machines had 16 MB of RAM. The NFS node had 16 MB of RAM, and the SPARC Server 690MP had 128 MB RAM. All these machines (except for the NFS node) were shared with normal University and Department computing.

During training in the spring semester, students began to submit multiple filters. This was OK as long as the filters ran on different RISC machines. But some students did not pay attention and they started submitting multiple filters on a RISC machine. Many of the filters were doomed because they were not set up properly. This caused severe network problems and it was difficult to even try to log into some of the RISC machines on many occasions. Other students started rebooting the RISC machines and this stopped many filter runs.

So, during the summer semester, only Dr. Driscoll could activate a filter run on the RISC machines. Students had to meet Dr. Driscoll and he had to approve their filter files before he would run them. This solved all the network and machine load problems, but required that Dr. Driscoll be available about 8 hours a day just to look at and run filters.

Performance Results and Analysis

For the Category A routing task, fifty filters were developed for one routing run (UCF100). A large amount of time went into the development of 28 filters. The other 22 filters were developed rather quickly in the last few weeks before we turned in our routing queries. For just one of the topics, our filtering approach had the best average precision. For 16 of the topics, our average precision was above the median; for 34 of the topics, our average precision was below the median. Our overall average precision was .2285 for this experiment.

This performance is a lot worse than our performance for the routing experiment last year. We believe the following contributed to this year's lower performance:

1. We did not use training data from the Vol. 3 CD.

2. The Ziff document collection on the Vol. 3 CD was part of the "new" routing document collection.

3. We believed a lot of the training data was incorrect and we made our filters retrieve what we believed to be relevant, not what the assessor considered relevant.

Not using the training data on the Vol. 3 CD was a mistake because it is probably the most accurate of the training data. We should have used it. But we believe the real problem is that combined with the fact that the Ziff document collection on the Vol. 3 CD was also part of the "new" routing document collection! We developed our filters from March to late July and sent in our routing queries during the last week of July. That is when we discovered that the Ziff document collection on the Vol. 3 CD was part of the "new" routing document collection. Topics in the range of Topic 051 through Topic 150 had training data on the Vol. 3 CD. For all but one of the routing topics chosen this year in that range, our filters performed below the median. The highest performance was for our Topic 117 filter which performed right at the median this year. We used our Topic 117 routing filter from last year without making a change to it; last year it had the best performance! So, we believe item 1 and item 2 above really hurt our performance. We also believe item 3 above was a factor causing our lower performance.

For the Spanish adhoc task, two runs were made. The UCFSP1 run was our primary run to be evaluated. The difference between the UCFSP2 run and the UCFSP1 run is explained in an earlier section. The performance of the UCFSP1 run is extremely good. It had the best average precision for 17 of the 25 topics. On all but one topic, Topic SP45, average precision was above the median. The overall average precision was .4918 for this experiment. The overall average precision of the UCFSP2 run was just slightly lower than that of UCFSP1; and, for most topics, the UCFSP2 performance was just below the UCFSP1 performance; however, UCFSP2 also had the best average precision for 3 more of the 25 topics. So, our filtering approach generated best average precision performance for 20 of the 25 Spanish adhoc topics.

Negative weights were used in fourteen of the twenty-five UCFSP2 filters, and generally lowered precision, when compared to corresponding UCFSP1 runs in which no negative weighting was used. For two topics, Topic SP37 and Topic SP41, higher precision was obtained using negative weights. The logic behind the negation used in Topic SP41 was explained earlier, and the negation used in Topic SP37 will be described in a later section, where Topic SP37 was chosen for a multilingual experiment. These mixed results suggest that in an adhoc environment, where no training is permitted, negation must be used very carefully, if at all.

Multiple patterns, used in seven of the UCFSP2 filters, were also generally unsuccessful. A higher precision was attained by the use of multiple patterns for only Topic SP50. In that particular case, the weighting of the corresponding UCFSP1 filter was far from optimum. The topic dealt with "The Fabrication of Gold and Silver Jewelry in Mexico." Higher weight should have been assigned to the synonym list for the entity "jewelry" than to the domain list associated with the jewelry attribute "type of metal." We later obtained a precision of above .8 for Topic SP50 by changing the weights used in the UCFSP1 filter submitted for Topic SP50. Our official precision for this topic was .3358 for UCFSP1 and .4750 for UCFSP2, both above the median, but somewhat lower than that of the best performing filter. It appears that poor weighting was a factor in most cases where our performance was not so good.

A Multilingual Experiment

After submitting our Spanish runs, we began reading some of the documents that we had retrieved as relevant to get some indication of our Spanish Adhoc performance. We noticed that there were many documents in the *El Norte* collection that were relevant to English topics. Since the filtering system had already been modified to accept multiple patterns, the idea occurred to us that possibly a single filter could be used to scan both the English and Spanish collections in one single run, producing a single ranked list.

The idea was extended to include test documents in French and German when an article relevant to Topic SP37 (Evidence of Aztec Heritage and Culture in Mexico) was found in the magazine *Quinto Lingo* [8]. The article had been translated into English, Spanish, French, and German versions. All four versions of the article, "Who was Quetzalcoatl?" were typed in <SGML> format as separate documents, and put into a file to be scanned along with the Volume 1 CD, the Volume 2 CD, and the *El Norte* collection. Accented and umlauted characters were typed as regular unaccented characters. It should be noted that no additional French or German text was scanned, mainly due to lack of time and a desire not to infringe in any manner on copyrights.

Since the UCFSP2 query for Topic SP37, which uses negative weights, had the best precision and recall for that topic, it was decided that the closest translations of that query that were possible, given our limited familiarity with French and German, would be used to create separate synonym and domain lists for each additional language. Small allowances were made for differences in verb conjugation, noun and adjective declension, gender, and the way in which possessives are formed. The German lists reflect a non-speaker's humble attempt to master the grammar of a relatively complex language in a few hours. Declension of German nouns and adjectives is incomplete. Possessives using *of,* and equivalent prepositions in other languages (*de*, *du*, *auf*, etc.) were not included, as that form of possessive was overlooked in the original Spanish query. Accents and umlauts were removed. See Appendices B-E for the actual synonym and domain lists that were used for the various languages.

A single INF file was created called "multi.inf." The version presented at NIST was the version we actually ran, which used DOS compatible file names. Here we present an easier to understand version:

```
37                              (topic number)
200                             (window size)
multi.out                       (the ranked list)
Aztec_culture.syn.Span          (Spanish synonym files)
Aztec.syn.Span
culture.syn.Span
Aztec_culture.syn.Eng           (English synonym files)
Aztec.syn.Eng
culture.syn.Eng
Aztec_culture.syn.Fren          (French synonym files)
Aztec.syn.Fren
culture.syn.Fren
Aztec_culture.syn.Ger           (German synonym files)
Aztec.syn.Ger
culture.syn.Ger
evidence.dom.Nah                (domain file in the Aztec language, Nahuatl)
garbage.dom.Span                (a list of Spanish words we don't want)
0.00                            (lowest weighting to be output)
4                               (number of patterns)
6 3 2 0 0 0 0 0 0 0 0 0 1 -3    (pattern for Spanish)
0 0 0 6 3 2 0 0 0 0 0 0 1 -3    (pattern for English)
0 0 0 0 0 0 6 3 2 0 0 0 1 -3    (pattern for French)
0 0 0 0 0 0 0 0 0 6 3 2 1 -3    (pattern for German)
```

As can be seen from the patterns, equivalent synonym lists are given the same weight for each language.

The Nahuatl words in "evidence.dom.Nah" are, for the most part, the names of Aztec, Toltec, and Mayan deities. They were put in a domain file because "deity" could be looked upon as a subset of a "type" attribute for the entity "evidence." "evidence.dom.Nah" was given weight in the patterns for all four languages because we presumed that there would be no attempt to convert these names to another language. We did expect some misspellings and we found some when reading the documents that we retrieved.

The "garbage.dom.Span" file is the same as used in the original Spanish query. It seeks to avoid documents pertaining to "Aztec Stadium," and "Aztec Avenue" (in Monterrey, Nuevo Leon, where *El Norte* is published). After reading some retrieved documents, we discovered that there is also an "Aztec Television" in Mexico, which we could have included in this negatively weighted list. We did not attempt to translate this list to other languages, but gave it the same negative weight in all patterns (if a phrase from this list showed up in a non-Spanish document it would still warrant a negative weight). We could have chosen to give it a zero weight in non-Spanish patterns.

We ran our multilingual filter across Vol. 1 and Vol. 2 CDs of the English collection, the *El Norte* collection, and the test file from *Quinto Lingo* in about six hours. The top 25 documents are as follow:

37 Q0 SP94-0000465	0	0.87533 UCFSP2
37 Q0 WSJ910417-0120	1	0.86645 UCFSP2
37 Q0 FRA000000-0001	2	0.86320 UCFSP2 (French document from *Quinto Lingo*)
37 Q0 SP94-0037330	3	0.86112 UCFSP2
37 Q0 SP94-0043006	4	0.86048 UCFSP2
37 Q0 SP94-0110606	5	0.86011 UCFSP2
37 Q0 SP94-0005842	6	0.73133 UCFSP2
37 Q0 AP881027-0004	7	0.72425 UCFSP2
37 Q0 AP881020-0253	8	0.72425 UCFSP2
37 Q0 SP94-0202852	9	0.72312 UCFSP2
37 Q0 SP94-0000857	10	0.72094 UCFSP2
37 Q0 SP94-0110560	11	0.72020 UCFSP2
37 Q0 AP890828-0214	12	0.72008 UCFSP2
37 Q0 AP890728-0053	13	0.71979 UCFSP2
37 Q0 SP94-0101980	14	0.71615 UCFSP2
37 Q0 ENG000000-0001	15	0.21872 UCFSP2 (English document from *Quinto Lingo*)
37 Q0 DEU000000-0001	16	0.21823 UCFSP2 (German document from *Quinto Lingo*)
37 Q0 ESP000000-0001	17	0.21790 UCFSP2 (Spanish document from *Quinto Lingo*)
37 Q0 SP94-0033589	18	0.21758 UCFSP2
37 Q0 SP94-0110291	19	0.21748 UCFSP2
37 Q0 SP94-0107105	20	0.21740 UCFSP2
37 Q0 AP880314-0254	21	0.21706 UCFSP2
37 Q0 SP94-0038957	22	0.21700 UCFSP2
37 Q0 SP94-0003010	23	0.21669 UCFSP2
37 Q0 WSJ910905-0078	24	0.21609 UCFSP2

As seen above, the four known relevant documents from *Quinto Lingo* all appeared in our top 25 documents. The French document received a higher rank than its counterparts due to a difference in the way a possessive form was expressed in the translation. The top 25 also included 14 Spanish documents, 13 of which were judged to be relevant, and 7 English documents, 6 of which we considered relevant. Of the 1000 documents that we ranked, 111 were from the Vol. 1 and Vol. 2 CDs. Although we did not take the time to read all of these documents, it is interesting to note that we had document identifiers from every directory on the Vol. 1 and Vol. 2 CDs in our ranked top 1000.

This experiment sidesteps some important aspects of the grammars involved, particularly verb conjugation, and lacks an adequate test collection of French and German documents. It does, however, serve to illustrate that our filtering system is very flexible. To accurately scan text in French or German it would be necessary to modify the filter somewhat to deal with special French accents, and an additional German character resembling *ß*. Auxiliary programs could be built to conjugate at least the regular verbs of both languages, and possibly to deal with some of the noun and adjective declension seen in German. Some care would have to be taken to avoid using words in lists for one language that would mean something else in another, or some method would have to be devised to automatically determine the language of a document, and which pattern to use. We feel that Italian and Portuguese could also be filtered with only a small amount of alteration to our system.

Acknowledgments

Sara Abbott performed all of the Spanish experiments, as well as the multilingual experiment. Kai-Lin Hu created 22 filters during the few weeks just before routing queries were submitted to NIST. In some awful confusion in August, some of her very good filters were accidentally not used for submission of final results.

Michael Miller studied the source code of our filter system and made himself available for questions about its operation. He also developed useful utility programs for evaluating filter runs and retrieving documents. Gary Theis is the original author of the filter system source code. For the routing experiments, the relevancy evaluation within the filter system was not changed from that developed by Gary.

The following students worked very hard developing filters for the routing topics: Fernando DaSilva, Richard Edmundson, Fritz Feuerbacher, Carl Free, Keith Hamelin, Brian Hinken, Hong Ji, John Kuhn, Stewart Ort, Gladys Otegbeye, Xiang Gen Qian, Ed Saab, Mark Tootle, and Ahmed Wreiden. Alicia Chea helped with the French synonym and domain files used in the multilingual experiment.

Chris Buckley provided the evaluation of one of Cornell's TREC-3 Spanish adhoc runs so that we would have a benchmark for initial experiments involving Topics SP1-SP25.

También queremos agradecer a nuestros amigos de la comunidad latina quienes han ofrecido sus consejos y nos han ayudado en varias otras maneras. Se incluyen a Bernardo Arteaga, José y Ilda Ayala, Luís Castañeda, Miguel de Jesús Ruíz Reyes, y Salvador Ruíz Reyes, todos del lindo país de México, asi como a Roger Huamán y Peggie Castle Haas, de Lima Perú. Sobre todo agradecemos a la Dra. Flor de María Gallardo Vásquez, también de Lima, por haber leído tantos documentos, dando su opinión sobre la pertinencia de todos. El éxito que tengamos con la colección "El Norte" lo debemos a estos queridos amigos.

References

[1] Martín Alonzo, *Diccionario de Sinónimos Explicados*, Madrid: EDAF, 1984.

[2] N. J. Belkin and W. Bruce Croft, "Information Filtering and Information Retrieval: Two Sides to the Same Coin?", *Communications of the ACM*, Vol. 35, No. 12, pp. 29-38, 1992.

[3] C. J. Date, *An Introduction to Database Systems (Sixth Edition)*, Addison-Wesley Publishing Company, Inc., 1995.

[4] J. Driscoll, G. Theis and G. Billings, "Using Database Schemas to Detect Relevant Information", *Proceedings of the Third Text Retrieval Conference (TREC-3)*, NIST Special Publication 500-225 (D. K. Harman, ed.), April 1995.

[5] R. Elmasri and S. B. Navathe, *Fundamentals of Database Systems (Second Edition)*, The Benjamin/Cummings Publishing Company, Inc., 1994.

[6] Robert Famighetti, ed., *The World Almanac and Book of Facts*, Funk & Wagnalls, pp.46-53, 1994.

[7] Ramón García-Pelayo y Gross, *Larousse Diccionario Escolar (First Edition)*, México 06600, D.F., Larousse, 1987.

[8] Lowell Harmer, "Who Was Quetzalcoatl?", *Quinto Lingo, The Multi-lingual Magazine*, Vol. 15, No. 1, American National Heritage Assoc., Arlington VA, September 1977.

[9] James L. Nolan etal., *Mexico Business: The Portable Encyclopedia for Doing Business with Mexico*, World Trade Press, San Rafael, CA, 1994.

[10] Salvatore Ramondino, ed., *The New World SPANISH-ENGLISH and ENGLISH-SPANISH Dictionary*, Signet, New York, 1991.

APPENDIX A

EXPANSION OF VERBS AND ADJECTIVES

(This list was developed to test the *lexpand* program on English Topic 135.)

A sample preliminary synonym list

ANALISIS,
ANALIZAR*,
CLARIFICAR*,
CLARIFICACION,
CONCLUIR*,
CONCLUSION,
CONCLUSIONES,
CURAR*,
CURACION,
CURACIONES,
DESARROLLAR*,
PROCESAR*,
DESCUBRIR*,
HALLAR*,
EVIDENCIAR*,
MOSTRAR*,
MOSTRACION,
DEMONSTRAR*,
DEMOSTRACION,
PROBAR*,
PROBACION,
EXPERIMENTAR*,
EXPERIMENTACION,
INTRODUCIR*,
INTRODUCCION,
LLEVAR*,
CONDUCIR*,
CONDUCCION,
INVESTIGAR*,
INVESTIGACION,
RECONOCER*,
ESCRUDIÑAR*,
INDAEGAR*,
SONDEAR*,
REVOLUCIONAR*,
REVOLUCIONACION,
ILUMINAR*,
ILUMINACION,
ALUMBRAR*,
ENFOCAR*,
ESTUDIAR*,
ESTUDIOS,
TECNICO*,
TECNOLOGO*,
TECNOLOGIA,
TECHNOLOGIAS,
EXAMINAR*,
EXAMEN,
EXAMENES,
EXAMINACION,
EXAMINACIONES,
TERAPIA,
TERAPIAS,
TRATAR*,
TRATAMIENTO,
TRATAMIENTOS,
COMPRENDER*,
ENTENDER*,
#

The verb *ANALIZAR
as expanded

ANALIZO,
ANALIZAS,
ANALIZA,
ANALIZAMOS,
ANALIZAIS,
ANALIZAN,

ANALIZABA,
ANALIZABAS,
ANALIZABAMOS,
ANALIZABAIS,
ANALIZABAN,

ANALICE,
ANALIZASTE,
ANALIZO,
ANALIZASTEIS,
ANALIZARON,

ANALIZARE,
ANALIZARAS,
ANALIZARA,
ANALIZAREMOS,
ANALIZAREIS,
ANALIZARAN,

ANALIZARIA,
ANALIZARIAS,
ANALIZARIAMOS,
ANALIZARIAIS,
ANALIZARIAN,

ANALICE,
ANALICES,
ANALICEMOS,
ANALICEIS,
ANALICEN,

ANALIZARA,
ANALIZARAS,
ANALIZARAMOS,
ANALIZARAIS,
ANALIZARAN,

ANALIZASE,
ANALIZASES,
ANALIZASEMOS,
ANALIZASEIS,
ANALIZASEN,

ANALIZAR,
ANALIZARSE,
ANALIZARSELO,
ANALIZARSELOS,
ANALIZARSELA,
ANALIZARSELAS,
ANALIZARME,
ANALIZARMELO,
ANALIZARMELOS,
ANALIZARMELA,
ANALIZARMELAS,
ANALIZARLO,
ANALIZARLOS,
ANALIZARLA,
ANALIZARLAS,
ANALIZARLE,
ANALIZARLES,
ANALIZARNOS,

ANALIZANDO,
ANALIZANDOSE,
ANALIZANDOSELO,
ANALIZANDOSELOS,
ANALIZANDOSELA,
ANALIZANDOSELAS,
ANALIZANDOME,
ANALIZANDOMELO,
ANALIZANDOMELOS,
ANALIZANDOMELA,
ANALIZANDOMELAS,
ANALIZANDOLO,
ANALIZANDOLOS,
ANALIZANDOLA,
ANALIZANDOLAS,
ANALIZANDOLE,
ANALIZANDOLES,
ANALIZANDONOS,

ANALIZADO,
ANALIZADOS,
ANALIZADA,
ANALIZADAS,

The adjective *TECNICO
as expanded

TECNICO,
TECNICOS,
TECNICA,
TECNICAS,

APPENDIX B

SYNONYM AND DOMAIN FILES REFERENCED BY SPANISH PATTERN OF MULTILINGUAL EXPERIMENT

(weights indicated beside each file name)

<evidence.dom.Nah> 1

NETZAHUALCOYOTL,
MOTECUHZOMA,
XOCOYOTZIN,
TLALOC,
NAHUATL,
TEMPLO MAYOR,
MOCTEZUMA,
XIPE,
CAMAXTLI,
CAMAXTLI-MIXCOATL,
MICTLAN,
MICTLANTECUHTLI,
MICTLANCIUATL,
COATLIQUE,
COYOLXAUHQUI,
HUIZILOPOCHTLI,
HUITZILOPOCHTLI,
XIPETOTEC,
MIXCOATL,
TEZCATLIPOCA,
TENOCHTITLAN,
OMETECHTLI,
XOCHIQUETZAL,
XOCHIPILLI,
XOCHIPILLI-CINTEOTL,
CINTEOTL,
MITOTE,
MITOTES,
MONTECZUMA,
MONTEZUMA,
QUETZALCOATL,
#

<garbage.dom.Span> -3

AVENIDA AZTECA,
AVE AZTECA,
AV AZTECA,
ESTADIO AZTECA,
#

<Aztec.syn.Span> 3

AZTECA,
AZTECAS,
AZTEQUISMO,
AZTEQUISMOS,
#

<culture.syn.Span> 2

CULTURA,
CULTURAS,
HERENCIA,
HERENCIAS,
CULTURAL,
CULTURALES,
HISTORIA,
HISTORIAS,
MITO,
MITOS,
MUSEO,
MUSEOS,
MITOLOGIA,
CODICE,
CODICES,
MONOLITO,
MONOLITOS,
TEMPLO,
TEMPLOS,
#

<Aztec_culture.syn.Span> 6

HERENCIA AZTECA,
CULTURA AZTECA,
INFLUENCIA AZTECA,
INFLUENCIAS AZTECA,
INFLUENCIAS AZTECAS,
INFLUYO AZTECA,
INFLUYOS AZTECA,
INFLUYOS AZTECAS,
HISTORIAS AZTECA,
HISTORIA AZTECA,
HISTORIAS AZTECAS,
TEMPLO AZTECA,
TEMPLOS AZTECA,
TEMPLOS AZTECAS,
MITO AZTECA,
MITOS AZTECA,
MITOS AZTECAS,
MITOLOGIA AZTECA,
MITOLOGIAS AZTECA,
MITOLOGIAS AZTECAS,
CODICE AZTECA,
CODICES AZTECAS,
CODICES AZTECA,
DEIDAD AZTECA,
DEIDADES AZTECA,
DEIDADES AZTECAS,
DIOS AZTECA,
DIOSES AZTECA,
DIOSES AZTECAS,
DIOSA AZTECA,
DIOSAS AZTECA,
DIOSAS AZTECAS,
#

APPENDIX C

SYNONYM AND DOMAIN FILES REFERENCED BY ENGLISH PATTERN OF MULTILINGUAL EXPERIMENT

(weights indicated beside each file name)

<*evidence.dom.Nah*> 1

NETZAHUALCOYOTL,
MOTECUHZOMA,
XOCOYOTZIN,
TLALOC,
NAHUATL,
TEMPLO MAYOR,
MOCTEZUMA,
XIPE,
CAMAXTLI,
CAMAXTLI-MIXCOATL,
MICTLAN,
MICTLANTECUHTLI,
MICTLANCIUATL,
COATLIQUE,
COYOLXAUHQUI,
HUIZILOPOCHTLI,
HUITZILOPOCHTLI,
XIPETOTEC,
MIXCOATL,
TEZCATLIPOCA,
TENOCHTITLAN,
OMETECHTLI,
XOCHIQUETZAL,
XOCHIPILLI,
XOCHIPILLI-CINTEOTL,
CINTEOTL,
MITOTE,
MITOTES,
MONTECZUMA,
MONTEZUMA,
QUETZALCOATL,
#

<*garbage.dom.Span*> -3

AVENIDA AZTECA,
AVE AZTECA,
AV AZTECA,
ESTADIO AZTECA,
#

<*Aztec.syn.Eng*> 3

AZTEC,
AZTECS,
#

<*culture.syn.Eng*> 2

CULTURE,
CULTURES,
HERITAGE,
HERITAGES,
CULTURAL,
HISTORY,
HISTORIC,
HISTORICAL,
LEGEND,
LEGENDS,
LEGENDARY,
MYTH,
MYTHS,
MYTHICAL,
MUSEUM,
MUSEUMS,
MYTHOLOGY,
MYTHOLOGIES,
CODEX,
CODICES,
MONOLITH,
MONOLITHS,
TEMPLE,
TEMPLES,
#

<*Aztec_culture.syn.Eng*> 6

AZTEC HERITAGE,
AZTEC CULTURE,
AZTEC INFLUENCE,
AZTEC INFLUENCES,
AZTEC HISTORY,
AZTEC LEGEND,
AZTEC LEGENDS,
AZTEC TEMPLE,
AZTEC TEMPLES,
AZTEC MYTH,
AZTEC MYTHS,
AZTEC MYTHOLOGY,
AZTEC MYTHOLOGIES,
AZTEC CODEX,
AZTEC CODICES,
AZTEC DEITY,
AZTEC DEITIES,
AZTEC GOD,
AZTEC GODS,
AZTEC GODDESS,
AZTEC GODDESSES,
#

APPENDIX D

SYNONYM AND DOMAIN FILES REFERENCED BY FRENCH PATTERN OF MULTILINGUAL EXPERIMENT

(weights indicated beside each file name)

<evidence.dom.Nah> 1

NETZAHUALCOYOTL,
MOTECUHZOMA,
XOCOYOTZIN,
TLALOC,
NAHUATL,
TEMPLO MAYOR,
MOCTEZUMA,
XIPE,
CAMAXTLI,
CAMAXTLI-MIXCOATL,
MICTLAN,
MICTLANTECUHTLI,
MICTLANCIUATL,
COATLIQUE,
COYOLXAUHQUI,
HUIZILOPOCHTLI,
HUITZILOPOCHTLI,
XIPETOTEC,
MIXCOATL,
TEZCATLIPOCA,
TENOCHTITLAN,
OMETECHTLI,
XOCHIQUETZAL,
XOCHIPILLI,
XOCHIPILLI-CINTEOTL,
CINTEOTL,
MITOTE,
MITOTES,
MONTECZUMA,
MONTEZUMA,
QUETZALCOATL,
#

<garbage.dom.Span> -3

AVENIDA AZTECA,
AVE AZTECA,
AV AZTECA,
ESTADIO AZTECA,
#

<Aztec.syn.Fren> 3

AZTEQUE,
AZTEQUES,
AZTECS,
AZTEC,
#

<culture.syn.Fren> 2

CULTURE,
CULTURES,
HERITAGE,
HERITAGES,
CULTUREL,
CULTURELS,
CULTURELLE,
CULTURELLES,
HISTOIRE,
HISTOIRES,
HISTORIQUE,
HISTORIQUES,
LEGENDE,
LEGENDES,
LEGENDAIRE,
LEGENDAIRES,
MYTHE,
MYTHES,
MYTHIQUE,
MYTHIQUES,
MUSEE,
MUSEES,
MYTHOLOGIE,
MYTHOLOGIES,
CODICE,
CODICES,
MONOLITHE,
MONOLITHES,
TEMPLE,
TEMPLES,
#

<Aztec_culture.Fren> 6

HERITAGE AZTEQUE,
CULTURE AZTEQUE,
INFLUENCE AZTEQUE,
INFLUENCES AZTEQUES,
HISTOIRE AZTEQUE,
HISTOIRES AZTEQUES,
LEGENDE AZTEQUE,
LEGENDES AZTEQUES,
TEMPLE AZTEQUE,
TEMPLES AZTEQUES,
MYTHE AZTEQUE,
MYTHES AZTEQUES,
MYTHOLOGIE AZTEQUE,
MYTHOLOGIES AZTEQUES,
CODICE AZTEQUE,
CODICES AZTEQUES,
DIEU AZTEQUE,
DIEUX AZTEQUES,
DEESSE AZTEQUE,
DEESSES AZTEQUES,
#

APPENDIX E

SYNONYM AND DOMAIN FILES REFERENCED BY GERMAN PATTERN OF MULTILINGUAL EXPERIMENT

(weights indicated beside each file name)

<evidence.dom.Nah> I

NETZAHUALCOYOTL,
MOTECUHZOMA,
XOCOYOTZIN,
TLALOC,
NAHUATL,
TEMPLO MAYOR,
MOCTEZUMA,
XIPE,
CAMAXTLI,
CAMAXTLI-MIXCOATL,
MICTLAN,
MICTLANTECUHTLI,
MICTLANCIUATL,
COATLIQUE,
COYOLXAUHQUI,
HUIZILOPOCHTLI,
HUITZILOPOCHTLI,
XIPETOTEC,
MIXCOATL,
TEZCATLIPOCA,
TENOCHTITLAN,
OMETECHTLI,
XOCHIQUETZAL,
XOCHIPILLI,
XOCHIPILLI-CINTEOTL,
CINTEOTL,
MITOTE,
MITOTES,
MONTECZUMA,
MONTEZUMA,
QUETZALCOATL,
#

<garbage.dom.Span> -3

AVENIDA AZTECA,
AVE AZTECA,
AV AZTECA,
ESTADIO AZTECA,
#

<Aztec.syn.Ger> 3

AZTEK,
AZTEKEN,
AZTEKER,
AZTEKERN,
AZTEKISCH,
AZTEKISCHE,
AZTEKISCHEN,
AZTEKISCHER,
AZTEKISCHERN,
#

<culture.syn.Ger> 2

KULTUR,
KULTUREN,
BILDUNG,
BILDUNGEN,
ERBSCHAFT,
ERBSCHAFTEN,
KULTURELL,
KULTURZENTRUM,
KULTURZENTREN,
GESCHICHTE,
GESCHICHTEN,
GESCHICHTSWISSENSCHAFT,
GESCHICHTSWISSENSCHAFTEN,
HISTORISCH,
HISTORISCHE,
HISTORISCHEN,
HISTORISCHER,
HISTORISCHERN,
GESCHICHTLICH,
GESCHICHTLICHE,
GESCHICHTLICHEN,
GESCHICHTLICHER,
GESCHICHTLICHERN,
LEGENDE,
LEGENDEN,
MYTHUS,
MYTHOS,
MYTHE,
MYTHEN,
MYTHERN,
SAGE,
SAGEN,
MUSEUM,
MUSEEN,
MYTHOLOGIE,
MYTHOLOGIEN,
KODEX,
KODEXEN,
MONOLITHE,
MONOLITHEN,
TEMPEL,
TEMPLEN,
#

<Aztec_culture.syn.Ger> 6

AZTEKISCHE ERBSCHAFT,
AZTEKISCHE KULTUR,
AZTEKISCH EINFLUB,
AZTEKISCHER EINFLUSSE,
AZTEKISCHE GESCHICHTE,
AZTEKISCHE GESCHICHTSWISSENSHAFT,
AZTEKISCHER TEMPEL,
AZTEKISCH TEMPEL,
AZTEKISCH MYTHOS,
AZTEKISCH MYTHUS,
AZTEKISCHE MYTHE,
AZTEKISCHER MYTHEN,
AZTEKISCHE LEGENDE,
AZTEKISCHEN LEGENDEN,
AZTEKISCHE MYTHOLOGIE,
AZTEKISCHEN MYTHOLOGIEN,
AZTEKISCH KODEX,
AZTEKISCHER KODEXE,
AZTEKISCH GOTT,
AZTEKISCHER GOTTER,
AZTEKISCHERN GOTTERN,
AZTEKISCHE GOTTIN,
AZTEKISCHEN GOTTINEN,
AZTEKISCHE SAGE,
AZTEKISCHEN SAGEN,
#

Similarity Measures for Short Queries

Ross Wilkinson Justin Zobel Ron Sacks-Davis
Department of Computer Science, RMIT
GPO Box 2476V, Melbourne 3001, Australia
{ross,jz,rsd}@cs.rmit.edu.au

October 1995

Abstract

Ad-hoc queries are usually short, of perhaps two to ten terms. However, in previous rounds of TREC we have concentrated on obtaining optimal performance for the long TREC topics. In this paper we investigate the behaviour of similarity measures on short queries, and show experimentally that two successful measures—which give similar, good performance on long TREC topics—do not work well for short queries. We explore methods for achieving greater effectiveness for short queries, and conclude that a successful approach is to combine these similarity measures with other evidence. We also briefly describe our experiments with the Spanish data.

1 Introduction

Users of interactive text retrieval systems often pose short queries, typically of a few words only. Our experience with users of real systems who pose such queries is that they are unsatisfied with the behaviour of standard similarity measures such as cosine, reporting that the answers often have little apparent correspondence to the query.

These problems are despite consistent positive results for such similarity measures on test databases, including the first three rounds of TREC experiments. One feature of the TREC experiments has, however, been the length of the topics, which are intended to be careful descriptions of information needs rather than ad-hoc queries. Problems that are specific to short queries did not emerge; in particular, with short queries there is a tendency for documents containing several occurrences of the rarest query term to be ranked first, regardless of whether the other query terms occur. It is not at all clear, therefore, that a similarity measure that is successful for TREC topics 1 to 200 will work well for short queries.

As part of other work, Zobel and Moffat have been using automatic (or "hill-climbing") techniques to identify effective similarity measures [6]. These experiments identified—for the full TREC topics—effective formulations of cosine, in broad terms confirming current wisdom about which formulations work well, but also confirming that the "Okapi" measure (or 2-Poisson method) [3, 4] is slightly more effective.

In this paper we explore similarity measures designed to address the problems of short queries. We show that when the cosine and Okapi measures are applied to short forms of the TREC topics, very different behaviour emerged, with the cosine measure in particular performing far worse. We propose that these measures be combined with other evidence, and that in particular it is advantageous to highly rank documents that contain all the query terms, even if they are not particularly highly ranked by the similarity measure. These conjectures are confirmed by our results in the current round of TREC experiments.

Hill-climbing is briefly summarised in Section 2. Approaches to similarity for short queries are discussed in Sections 3 and 4, which also gives the results of TREC experiments for this year. Initial experiments with the Spanish data are reported in Section 5. We conclude in Section 6.

2 Hill climbing

There has been continued refinement of similarity measures such as cosine, with several changes over recent years that result in better performance: use of logs in within-document frequency, modification to term weighting in queries, and so on. These changes not only impact on effectiveness, but have implications for efficient implementation.

Zobel and Moffat have, as part of separate work, been using automatic methods to search for new similarity measures [6], with the primary aim of identifying effective techniques that can be evaluated efficiently. Given the number of different proposed formulations for similarity measures, and of formulations of their components such as term weight, it is quite possible that some more effective formulation is as yet undiscovered; and equally possible that there is some formulation that is not only effective, but easier to evaluate. Standard cosine requires, for example, a table of document weights based on collection-wide information, and a cosine that did not require this table could be evaluated more efficiently.

The technique used by Zobel and Moffat was to break similarity measures into several components: combining function, relative term frequency, global term weight, in-document and in-query frequency, and document and query length. Several formulations were made available for each component, giving 311,040 combinations, of which about 70,000 are mathematically distinct. These formulations were incorporated into the MG text retrieval system [1]. Using WSJ and the relevance judgements for topics 51 to 200 (and using AP in separate experiments to confirm the results), effective formulations were found by the hill-climbing method of search for the best variant of each combination of components. Interestingly—and vindicating the value of this approach—the most successful formulations were not precisely those expected. In particular, the standard formulation of document length was no more effective than using approximated lengths such as the square root of the number of distinct terms in the document.

The hill-climbing approach identified many variants of the cosine measure that were about equally effective. We chose one of these for use in our experiments, as shown in the next section. Slightly more effective was a variant on the Okapi measure, also shown in the next section.

3 Short queries versus long queries

Users tend to issue short queries when examining text databases online. Queries may be extended with relevance feedback, or queries by example may be developed during a query session. Yet the first query can be crucial, perhaps the most important in a retrieval session. However in most test collections, and notably in TREC, the supplied queries have been long, and experimental systems have been optimised with regard to these queries.

Test	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%	Av.
CDM	0.464	0.218	0.143	0.096	0.069	0.047	0.029	0.008	0.008	0.002	0.000	0.078
WCM	0.625	0.404	0.309	0.241	0.193	0.145	0.112	0.046	0.046	0.021	0.004	0.177
COS	0.803	0.607	0.483	0.395	0.327	0.262	0.210	0.092	0.092	0.040	0.006	0.286
OKA	0.770	0.537	0.417	0.331	0.271	0.218	0.166	0.074	0.074	0.033	0.006	0.244

Table 1: Precision of similarity measures for full TREC topics (WSJ)

In feedback from users of the Structured Information Manager SIM, a commercial SGML-conformant document management system developed at CITRI [5], we have found that people find it more acceptable to see documents that have many matching terms, compared to one term matching many times. This is true even if the document is irrelevant. [1] That is, they would find some kind of *coordination match* as an important component of any similarity measure.

We considered four measures of similarity in these investigations:

Coordination match:

$$\mathrm{CDM}(q,d) = |\{t : t \in q \wedge t \in d\}|$$

Weighted coordination match:

$$\mathrm{WCM}(q,d) = \sum_{t \in q \wedge t \in d} \log(N/f_t)$$

Cosine measure:

$$\mathrm{COS}(q,d) = \frac{\sum_{t \in q \wedge t \in d} (w_{q,t} \cdot w_{d,t})}{\sqrt{\left(\sum_{t \in q} (w_{q,t})^2 \cdot \sum_{t \in d} (w_{d,t})^2\right)}}$$

A variant of the Okapi measure:

$$\mathrm{OKA}(q,d) = \sum_{t \in q \wedge t \in d} \log\left(\frac{N - f_t}{f_t}\right)\left(\frac{f_{d,t}}{f_{d,t} + \sqrt{f_d}/av(\sqrt{f_d})}\right)$$

where $w_{q,t} = log(N/f_t + 1)$ and $w_{d,t} = log(f_{d,t} + 1)$, $f_{x,t}$ is the frequency of t in x, f_d is the number of terms in document d, N is the number of documents in the collection, and f_t is the number of documents containing t.

We used topics 51 to 200 against the Wall Street Journal sub-collection of the TREC collection to guide our investigations. We first give the results for each of these measures using standard stopping and stemming of terms. No special processing of the data to find index terms was used in any of these experiments. The top 1000 ranked documents were used for analysis. Table 1 gives effectiveness for each of the above similarity measures on these topics. We can see the clear advantage of the two more sophisticated measures over the co-ordination match measures. This is to be expected of long queries.

[1] Thus there is a human-factors question about whether retrieval effectiveness is the appropriate property to optimise.

Query set	Terms
TREC topics	77
Titles	4
Hand	6

Table 2: Query lengths

Test	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%	Av.
						Titles						
CDM	0.400	0.222	0.182	0.139	0.120	0.097	0.079	0.034	0.034	0.023	0.009	0.102
WCM	0.420	0.259	0.217	0.182	0.159	0.131	0.111	0.074	0.074	0.054	0.016	0.133
COS	0.531	0.379	0.300	0.255	0.211	0.164	0.119	0.050	0.050	0.026	0.004	0.175
OKA	0.706	0.514	0.414	0.331	0.261	0.192	0.148	0.055	0.055	0.031	0.009	0.230
						Hand						
CDM	0.502	0.330	0.262	0.214	0.180	0.144	0.099	0.039	0.039	0.022	0.009	0.147
WCM	0.525	0.371	0.322	0.274	0.233	0.182	0.153	0.090	0.090	0.059	0.020	0.191
COS	0.652	0.471	0.395	0.333	0.270	0.215	0.164	0.067	0.067	0.032	0.006	0.226
OKA	0.776	0.598	0.512	0.434	0.371	0.310	0.252	0.138	0.138	0.076	0.018	0.316

Table 3: Precision of similarity measures for short queries 51-200 (WSJ)

To investigate behaviour on short queries, a source of short queries is required. One possibility is to use the titles of the TREC topics. Since it was not clear that titles field were intended to stand alone as queries, we also created with our own short query statement for each topic, which was typically a list of the most important keywords. The average number of terms in each form of the query sets is given in Table 2. The results of using these short query expressions are given in Table 3.

We see in these experiments that coordination match approaches do not work well in comparison to more sophisticated methods. WCM did however identify a larger pool of relevant documents. It is also interesting to note how much better the Okapi measure did than the cosine measure for short queries. It seems clear that there is potential for taking some account of coordination match. We look below at a number of ways of combining evidence.

We found another telling reason for looking at some form of combination: each of the methods are producing significantly different collections of high-ranking documents. When the topics 201–250 where evaluated against the TREC collection and the top 1,000 documents retrieved for each of CDM, COS, OKA, we examined how many documents were ranked highly by pairs of methods. If the measures were identical, we would expect the overlap of the sets of answers of the different methods to be 50,000 documents. If they were completely incompatible, we would get an overlap of 0 documents.

We found that the overlap between CDM and COS was 9,106, that between CDM and OKA was 21,101, and remarkably, that between COS and OKA was only 10,561. Intuitively we would thus expect that combining evidence would help by identifying those documents found by all methods.

Test	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%	Av.
COS	0.652	0.471	0.395	0.333	0.270	0.215	0.164	0.067	0.067	0.032	0.006	0.226
(CDM>COS)	0.724	0.512	0.423	0.345	0.283	0.213	0.148	0.066	0.066	0.032	0.014	0.235
(COS>CDM)	0.726	0.525	0.443	0.367	0.307	0.241	0.191	0.088	0.088	0.045	0.013	0.257
(WCM>COS)	0.701	0.487	0.409	0.337	0.281	0.215	0.176	0.101	0.101	0.063	0.021	0.242
(COS>WCM)	0.711	0.510	0.439	0.367	0.309	0.250	0.211	0.103	0.103	0.049	0.010	0.260
(CDM+COS)	0.727	0.526	0.444	0.367	0.307	0.241	0.191	0.088	0.088	0.045	0.013	0.258
(WCM+COS)	0.733	0.526	0.454	0.383	0.318	0.259	0.216	0.103	0.103	0.050	0.010	0.270
r(WCM+COS)	0.727	0.533	0.463	0.404	0.337	0.279	0.229	0.121	0.121	0.070	0.017	0.285

Table 4: Precision of similarity measures for combined evidence

4 Combined approaches

How might we combine the evidence of the different similarity measures? If we accept the user's view that the document with the most matches is best, we simply order by coordination match, and then use the cosine measure to order documents with the same number of terms in common. In subsequent results in this section we give results for topics 51–200 with the hand-generated short queries against the Wall Street Journal Collection. We see in Table 4 that this approach (CDM>COS) gives a small improvement—of 4% over using the cosine measure alone.

The (CDM>COS) approach was applied only to the top 1,000 documents ranked by CDM. It is possible to use the corresponding approach of reordering the top 1,000 documents ranked by COS according to coordination match. This experiment, (COS>CDM), gave a further improvement of 9.7% simply due to the better pool of documents that were selected. Note that, should all documents in the database be ranked, then the results would be identical. However this method of combination provides a 13.7% improvement compared to COS for the set of short queries.

The weighted coordination match WCM gave better results than the unweighted coordination match, so we could combine in the same way and look for further improvements. It was possible, however, that improvements would not be available as this weighting is already provided by the cosine measure. Nevertheless (WCM>COS), the WCM equivalent of (CDM>COS), and (COS>WCM), the WCM equivalent of (COS>CDM), gave small improvements as shown in Table 4 .

Instead of using this two-tiered approach to ranking documents, we could regard the coordination match and the cosine measure simply as two different pieces of evidence that documents are relevant. In this case, we simply add the evidence in some way. Since the cosine measure takes values between 0 and 1, and the coordination match gives integer values, the measures had to be normalised, or the result of addition would simply be (CDM>COS) or (WCM>COS). One possibility is to divide the coordination match figures by the maximum possible value–that is, the number of terms in the query. Another possibility is to normalise each piece of evidence by dividing by the score of the highest-ranked document. In both cases, there remains the question of what relative weights to apply to the two pools of evidence. Linear regression is the obvious way of providing relative weights. (CDM+COS) gives the result of normalising the coordination match, and then adding to the cosine score, and (WCM+COS) gives the result with the weighted coordination match. r(WCM+COS) gives the result of normalising both the

Test	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%	Av.
CDM	0.270	0.094	0.058	0.033	0.020	0.011	0.007	0.001	0.000	0.000	0.000	0.028
WCM	0.381	0.224	0.176	0.155	0.121	0.099	0.064	0.046	0.031	0.016	0.005	0.100
COS	0.612	0.362	0.268	0.220	0.175	0.119	0.073	0.033	0.011	0.003	0.001	0.147
OKA	0.595	0.384	0.320	0.261	0.213	0.171	0.119	0.070	0.031	0.010	0.003	0.178
(WCM+COS)	0.668	0.422	0.336	0.278	0.234	0.160	0.100	0.065	0.038	0.015	0.003	0.187
(WCM+OKA)	0.570	0.328	0.281	0.236	0.185	0.151	0.106	0.059	0.040	0.016	0.005	0.156
(OKA+COS)	0.648	0.447	0.356	0.287	0.238	0.179	0.116	0.066	0.029	0.009	0.002	0.196

Table 5: Precision of similarity measures for on TREC queries 202–250

cosine and the weighted coordination match, and then regressing against the relevance results. While this last experiment is not in practice a valid way of combining evidence, since it can only be applied a postiori, it does give an indication of the maximal results achievable using this approach.

The next stage of experimentation involved using the Okapi measure instead of the cosine measure in combination with coordination measures. All of these experiments failed to give any improvement over using the Okapi measure on its own. This result was quite surprising. However, the cosine measure and Okapi have very different behaviour on small queries that helps explain the behaviour.

Suppose that we have two documents, A and B. Document A has a rare term that matches the query 4 times. Document B has two more common terms that match once each. Assuming that A and B are of average size, $N = 10^7$, $f_{t1} = 10$, and $f_{t2} = 100$, we find that

$$OKA(Q, A) \simeq log(10^6) \times 0.8 \simeq 11$$

$$OKA(Q, B) \simeq 2 \times log(10^5) \times 0.5 \simeq 11.5$$

By contrast,

$$COS(Q, A) \simeq log(10^6) \times log(5)/D \simeq 22.2/D$$

$$COS(Q, B) \simeq 2 \times log(10^5) \times log(2)/D \simeq 12.8/D$$

Thus the Okapi measure is much more like the coordination match than the cosine is. It thus became appropriate to regard the Okapi measure as a form of co-ordination match.

Topics 201–250 are much shorter than the previous TREC topics, at about 9 terms each. This size is still larger than we believe that people usually ask, and we did not expect that pure coordination match approaches were likely to be helpful. Thus we decided to use the Okapi measure, which had proved robust on smaller queries as our baseline. This was our CITRI1 run. We combined the Okapi measure and the cosine measure for our second run, to allow us to see whether the fact that these two measures produce very different result sets could be combined with good effect. The result of adding normalised Okapi scores to cosine scores are given in (OKA+COS) in Table 5, and constitute our CITRI2 run. We see that there is a significant gain in combining these two measures.

However, we are really interested in what happens with very short queries. Thus the topics 202–250 were again hand edited into queries with just 3–4 terms. The experiments were run against the Wall Street Journal database. The results are given in Table 6. It is remarkable how badly the cosine measure does on this data.

Test	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%	Av.
CDM	0.374	0.242	0.178	0.132	0.102	0.087	0.074	0.020	0.020	0.008	0.006	0.096
WCM	0.386	0.244	0.182	0.143	0.132	0.101	0.094	0.032	0.032	0.016	0.011	0.107
COS	0.226	0.177	0.127	0.089	0.068	0.055	0.044	0.013	0.013	0.008	0.004	0.066
OKA	0.497	0.337	0.242	0.196	0.146	0.116	0.094	0.033	0.033	0.019	0.013	0.139
(WCM+COS)	0.342	0.272	0.210	0.158	0.134	0.106	0.090	0.031	0.031	0.015	0.009	0.114
(WCM+OKA)	0.503	0.332	0.238	0.194	0.132	0.100	0.093	0.041	0.041	0.018	0.013	0.135
(OKA+COS)	0.361	0.273	0.223	0.158	0.124	0.103	0.084	0.025	0.025	0.016	0.012	0.116

Table 6: Precision of similarity measures for short queries 202-250 (WSJ)

Test	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%	Av.
S	0.851	0.657	0.542	0.435	0.374	0.314	0.246	0.095	0.095	0.022	0.000	0.315
SS	0.864	0.630	0.547	0.437	0.366	0.303	0.249	0.085	0.085	0.038	0.000	0.315
SSP	0.818	0.625	0.527	0.443	0.379	0.319	0.258	0.099	0.099	0.030	0.000	0.318
(S+SSP)	0.866	0.677	0.571	0.461	0.400	0.340	0.272	0.117	0.117	0.027	0.000	0.334

Table 7: Spanish Experiments (Queries 1–25)

5 Spanish experiments

In this, our first attempt at the Spanish track, we were principally interested in how well the indexing approaches that are used in English work in Spanish. Thus we developed a simple stop-list and a stemmer. However we were also interested exploring the notion of Church [2], that standard stemming approaches are too strong as normalisation processes. We felt that by looking at results where the data is both stemmed and not stemmed we might be able to gain, continuing our theme of combining evidence.

A stop-list of 316 words was created, and we used a stemmer that recursively removed 30 word endings. The suffixes were principally the regular verb suffixes. We then implemented an adjacent pair finder, with the constraint that the combined length of the stems was less than 10 characters. (This was to reduce the number of index terms to something less than a million terms.)

We were interested to see how we might combine the results from experiments where we stopped (S), stopped and stemmed (SS), and stopped, stemmed and paired (SSP). The results on queries 1–25 are given in Table 7. We looked at several ways of combining: S with SS, SS with SSP, S with SSP, and finally S with both SS and SSP. The best of these results was S with SSP, or (S+SSP). Thus our submitted runs are a run with just stopped data (CITRI-SP1), and then a run with the stopped results combined with stopped, stemmed and paired (CITRI-SP2). The results on queries 26–50 are given in Table 8.

Given the success of the Okapi measure, we give the same results using this measure in Table 9. We note that these experiments seem to indicate that a similar methodology to English works quite well for Spanish text.

Test	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%	Av.
s	0.595	0.337	0.263	0.206	0.165	0.121	0.081	0.014	0.014	0.000	0.000	0.143
ss	0.674	0.383	0.282	0.211	0.169	0.132	0.083	0.017	0.017	0.000	0.000	0.157
ssp	0.649	0.407	0.286	0.205	0.165	0.121	0.080	0.014	0.014	0.001	0.000	0.155
(s+ssp)	0.642	0.377	0.295	0.222	0.178	0.132	0.089	0.019	0.019	0.001	0.000	0.158

Table 8: Spanish Experiments (Queries 25–50)

Test	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%	Av.
s	0.725	0.398	0.302	0.238	0.201	0.161	0.116	0.041	0.041	0.016	0.000	0.178
ss	0.834	0.456	0.331	0.274	0.212	0.178	0.141	0.057	0.057	0.020	0.000	0.206
ssp	0.740	0.435	0.319	0.263	0.205	0.167	0.135	0.053	0.053	0.016	0.001	0.195
(s+ssp)	0.698	0.455	0.323	0.266	0.225	0.188	0.137	0.045	0.045	0.017	0.001	0.202

Table 9: Spanish Experiments (Queries 25–50) with Okapi Measure

6 Conclusions

We have investigated similarity measures for short queries. The preliminary results of our investigation are that some form of coordination match is important to information retrieval with such queries, but that coordination match should not be the primary factor determining the order of answers. Our results also confirm those of the City University group last year, that the Okapi method gives an excellent similarity measure.

It is important to note that we have seen very different performance from different measures as the average query length is varied. The coordination measures appear to introduce new useful documents. Different measures are finding very different result sets, so there is reason to believe that good improvement is possible.

Acknowledgements

This work was supported by the Australian Research Council. Descriptions of the systems used for this work are found at `http://www.kbs.citri.edu.au/sim/sim.html` and `http://www.kbs.citri.edu.au/mg`.

References

[1] T.C. Bell, A. Moffat, I.H. Witten, and J. Zobel. The MG retrieval system: Compressing for space and speed. *Communications of the ACM*, 38(4):41–42, April 1995.

[2] K. Church. One term or two? In *Proc. ACM-SIGIR International Conference on Research and Development in Information Retrieval*, pages 310–318, Seattle, Washington, July 1995.

[3] S.E. Robertson and S. Walker. Some simple effective approximations to the 2-Poisson method for probabilistic weighted retrieval. In W.B. Croft and C.J. van Rijsbergen, editors,

Proc. ACM-SIGIR International Conference on Research and Development in Information Retrieval, pages 232–241, Dublin, Ireland, July 1994.

[4] S.E. Robertson, S. Walker, S. Jones, M.M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-3. In D.K. Harman, editor, *Proc. Text Retrieval Conference (TREC)*, pages 109–126, Gaithersburg, Maryland, 1994. NIST Special Publication 500-225.

[5] R. Sacks-Davis, T. Arnold-Moore, and A. Kent. A standards-based approach to combining information retrieval and database functionality. International Journal of Information Technology, Vol. 1, No. 1, pages 1–15, 1995.

[6] J. Zobel and A. Moffat. Similarity measures explored. Technical Report TR-95–3, Collaborative Information Technology Research Institute, RMIT and The University of Melbourne, 1995.

InTEXT Precision Indexing in TREC4

Mark Burnett, Craig Fisher and Richard Jones

InTEXT Systems,

P.O. Box 310 Deakin West

ACT 2600 Australia
100032.1075@compuserve.com

A. Background

InTEXT Systems is a subsidiary of the CP Software Group, headquartered in Folsom California. The company is a leader in the provision of advanced tools and end-user products for intelligent document management. The InTEXT Research and Development group, based in Canberra Australia, has been in existence for 10 years, and develops leading edge technology in the area of text retrieval, indexing, routing, content analysis and document clustering, using their Heuristic Learning architecture.

The InTEXT Systems R and D group participated in TREC-1, when it formed the Centre for Electronic Document Research (CEDR), in conjunction with CITRI (Collaborative Information Technology Research Institute). [1]

B. Objectives

The principal objective of the experiment INTXT2, which was carried out within the AdHoc (manual) query section of TREC-4, was to establish the effectiveness of the index term selection techniques developed for the InTEXT Precision product.

Unlike most developments in full text retrieval, which concentrate entirely on methods to obtain optimal retrieval from a document collection where all words except a global stop-word list are included as index terms, Precision focuses on improving the process at the indexing stage by being more selective in the words chosen for indexing. In effect Precision selects a stop-word list per document. As the name suggest, a primary objective of the exercise is to improve the precision of the retrieval, it is hoped without significantly reducing the recall.

The objective of the INTXT2 experiment is to establish the effectiveness of the Precision methods for full text retrieval, using a commercially available text retrieval engine (InTEXT

Retrieval Engine) as the retrieval vehicle. In particular, it was hoped that considerably reduced hit lists would be obtained with high precision.

C. The Experiment

The original text files were split into documents, and all fields except the document reference and the full text were stripped. These were run through InTEXT Precision and two derived documents were created from each original. These were:

a PreciseScope document conforming to the markup for InTEXT Retrieval Engine with stop words inserted to replace the words not to be indexed;
a Keyword file of keys and associated weights. This was not used in the experiment.

A fragment of an original TREC document together with the two forms of Precision output is shown in Appendix 1. The complete set of PreciseScope documents and weighted keywords are available on request.

Text indexing and retrieval was performed by InTEXT Retrieval Engine, a commercially available full-text retrieval engine which ranks retrieved documents by heuristics that take into account the number and position of search terms within a document and compares these against a system-determined judgment of relevance for these terms.

The queries were created manually, using the TREC test document set with additional research at the National Library of Australia. It was found that a substantial knowledge of USA current affairs was required to understand the implication of some of the queries.

D. The InTEXT Precision Method

The Precision indexing approach uses the techniques employed in the InTEXT Systems Heuristic Learning Architecture, combining elements from:

identification of document surface structure: headings, paragraphs;
location of the author's message: cue phrases, repetition;
Learning from collection characteristics: frequency, discriminating power.

Term Generation: Precision operates by generating a large set of phrases, making use of a large stop word list (some 2,000 words). Grammatical variants are combined using a modified Lovins algorithm [2] and a score assigned to each phrase. This score is computed based upon the frequency and length of each phrase and its position within the document (in heading, first sentence of paragraph etc.) The combined phrases are ordered by their score, and the sum of the scores of the keys defines the Total Information Content (TIC) of the document.

Term Selection: Three classes of document are considered. For documents with a very low TIC, it is recognised that the Precision method does not work effectively and all non-stop words are indexed. A threshold was chosen for this experiment such that very few documents fell into this class. (Typically documents containing <200 words in total). Below the second threshold (about 500 words), all generated terms are used whose combined score adds up to a proportion of the TIC (80% for the experiment). Above this threshold, preference is given to terms that are noun like, particularly proper nouns, again up to a selected proportion of the document TIC.

PreciseScope Document creation: The selected terms including variants are mapped back onto the original document. Noise words are inserted in place of the other words so the positions of the selected words and phrases are correctly preserved. In this way, collocation searches (in same paragraph, within x words) work correctly. The process may be viewed as a dynamic selection of stop words for each document, defined as those which are not important in the document for defining its information content.

The process typically results in 10-20% of the words of the document being indexed. The whole process, including reading and writing all files, processed 30-40 Mbytes of source documents an hour on a 90Mhz Pentium.

The principal benefits of the Precision approach outlined above include:

- much smaller "hit lists" with higher precision, this is key in large document collections. No-one wants to read 1,000 documents in the hope that a few are relevant;
- quicker resolution of information requests;
- smaller index files and quicker generation and update of textbases.

The technique gives the potential for a new generation of full text retrieval, focusing on automatic index term selection, rather than purely on relevance measurement at query time. It is in principle compatible with any existing full text retrieval system.

E. Results of the Experiments

Across all documents some 225K different words were indexed, with a total word instance count of approximately 48M [3].

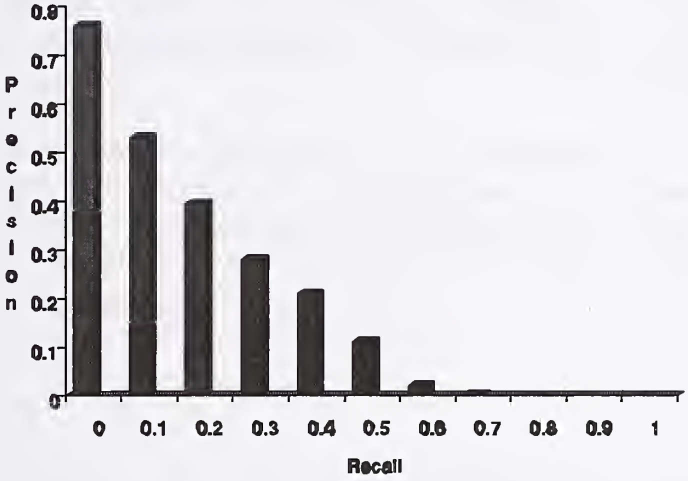
Some of the basic statistics across all queries are as follows:

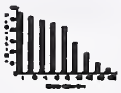
Total number of documents retrieved over all queries:	8992
Total number of relevant documents in collection:	6501
Total number of relevant and retrieved documents:	2161

- The interpolated Recall-Precision averages across all queries are shown in Figure 1.
- Average precision (non-interpolated) over all relevant documents is 0.1814
- The Precision averages across all queries are given in Figure 2.
- R-Precision (precision after R (= num_rel for a query) docs retrieved):
  Exact: 0.2615

Figure 1 - INTXT2 Precision/ Recall all Queries

Figure 2 - INTXT2 Precision/ Relevant Documents all Queries





F. Analysis

When viewed in the light of the experiment objectives, the results are very encouraging. 25% of retrieved documents are relevant and 33% of documents adjudged relevant were retrieved.

It is interesting to note that the precision recall figures are almost identical to those obtained by the InTEXT Routing experiments in TREC-1 [1]. The methods used for determining relevance are very similar to those in InTEXT Retrieval Engine. The difference is of course that in the current experiment, many fewer index terms (one in eight) are being utilised.

At a more detailed level, probably the most favourable statistics were obtained for the number of relevant retrieved at 100. In four queries the result obtained was the highest, in one of these 95 relevant documents were identified. The mean Precision figure for the top 30 retrieved documents across all queries was 0.42 (see Figure 2), a particularly good result. In no statistic was the INTXT2 result the worst!

On detailed analysis of some of the more disappointing queries, it seems that a major reason for poor performance was omission of search terms in the query rather than failure to select important terms for indexing. This is important given the objective of the experiment, and can clearly be addressed by more powerful query term selection methods, such as thesaurus use or automatic term expansion.

G. Conclusion

It is claimed that on the basis of the results of the INTXT2 experiment, the use of PreciseScope documents is a viable way of automatically indexing documents for full text retrieval, particularly where precision is a priority. As mentioned earlier, the InTEXT Precision output from this experiment is available on request

References

[1] R.L. Jones, S.K. Leung, D.L. Pape, "Application of the Automatic Message Router to the Tipster Collection", The First Text REtrieval Conference (TREC-1), National Institute of Standards and Technology, Special Publication 500-207, Gatithersburg, MD. 20899

[2] J. Lovins, "Development of a Stemming Algorithm", Mechanical Translations and Computational Linguistics, <u>11</u>, (1968), pp11-32

[3] J. Zobel, RMIT, Melbourne, Private communication

Appendix 1 Example Precision Documents

Original Document

```
AP901221-0008
<DOC>
 ............
 <TEXT>
Researchers in California and Utah report, in studies published today, new advances toward
isolating the gene linked to a susceptibility to breast cancer in some families.
    .............
Skolnick and King said their studies could lead eventually to tests enabling women with known
breast cancer risks, based on their family histories, to be treated even before the smallest of
tumors have formed.
     ........
Science, which published the studies, is the journal of the American Association for the
Advancement of Science.
</TEXT>
</DOC>
```

PreciseScope Document

```
$$T
AP901221-0008
    .........
... a10 STUDIES a01 a02 ... a13 BREAST CANCER a01 a02 FAMILIES
$$P
    ........
SKOLNICK a01 KING a01 a02 STUDIES a01 a02 ... a09 BREAST CANCER RISKS a01 a02
$$P
    .........
... a04 SCIENCE a01 a02 a03 STUDIES a01 a02
$$P
... a11 SCIENCE
$$N ENDTEXT
```

Precision Weighted Keys

```
AP901221-0008
$$N KEYS
    #breast#cancer=2050.6
    #breast#cancer#risk=926.0
    #proliferative#breast#disease=294.5
    #breast#cancer#gene=276.5
    #inherited#risk#of#breast=211.1
    #cancer#risk= 71.2
    #King= 64.0
    #Skolnick= 36.0
    #Science= 25.0
    #PBD=  9.0
    $$N ENDKEYS
```

Shortest Substring Ranking
(MultiText Experiments for TREC-4)

Charles L. A. Clarke   Gordon V. Cormack   Forbes J. Burkowski

MultiText Project
Department of Computer Science
University of Waterloo
Waterloo, Ontario, Canada
mt@plg.uwaterloo.ca

Abstract

To address the TREC-4 topics, we used a precise query language that yields and combines arbitrary intervals of text rather than pre-defined units like words and documents. Each solution was scored in inverse proportion to the length of the shortest interval containing it. Each document was scored by the sum of the scores of solutions within it. Whenever the above strategy yielded less than 1000 documents, documents satisfying successively weaker queries were added with lower rank. Our results for the ad-hoc topics compare favourably with the median average precision for all groups.

1 Introduction

The central concern of the MultiText project at the University of Waterloo is the management of data in large-scale distributed text database systems [10]. A major component of this work has been the development of a query language that is suitable for expressing queries over the heterogeneous data that is present in a very large text database. The query language developed for the MultiText project, called GCL, provides for the general expression of containment and ordering relationships between document components, phrase searching, and boolean searches solved independently of containment in documents or other predetermined components. A solution to a GCL query is an interval of text, which may or may not correspond to a document component.

Prior to TREC-4, development of GCL focused on the properties of the language that provide precise and simple query semantics and flexible retrieval from structured text. For our TREC-4 experiments we have addressed a different issue, focusing on how the properties of the language might be exploited to rank documents, or components of documents, when queries are expressed primarily in a boolean-flavored subset of GCL.

2 Overview of the GCL Query Language

In a traditional text database system, a simple boolean query

```
"future" AND ("vision" OR "prediction")
```

selects documents that satisfy a boolean expression, in this case by containing the term "future" and one of the terms "vision" or "prediction". In GCL the equivalent boolean expression is not solved with respect to any pre-defined component. Instead, the results of a query are simply the smallest intervals of text that satisfy the expression.

Within GCL, the results of a query may be used to select elements from the results of a second query using containment relationships, allowing queries to be solved in terms of arbitrary document components. For example, the GCL query:

```
P2 = "<paragraph>" ... "</paragraph>"  ... "</paragraph>"
```

specifies pairs of paragraphs and names the result "P2". The ordering operator "..." associates the start and end tags for the paragraphs that are presumed to exist in the text. Combining these queries with the GCL "containing" operator selects pairs of paragraphs that satisfy the boolean expression:

```
P2 containing ("future" AND ("vision" OR "prediction"))
```

Other GCL operators express other containment relationships: *contained in*, *not containing* and *not contained in*. All GCL operators are completely general and orthogonal; any query may be used as an operand to any operator. The query

```
("future" AND ("vision" OR "prediction")) contained in P2
```

returns intervals of text that satisfy the boolean expression and are contained within two paragraphs. GCL can search for phrases as well as single terms. The more unusual query

```
"luck of the draw" AND P2
```

finds intervals of text containing the phrase "luck of the draw" along with two paragraphs close-to or containing it, depending on whether the phrase appears in a paragraph.

A complete definition and discussion of GCL and its features appears elsewhere [4]. A precise formal semantics and implementation framework for GCL is also available [3].

3 Shortest Substring Solution Model

We treat the text in the database as a continuous sequence of *terms*, or *tokens*, each corresponding to a word or number in the text. Each term is assigned an integral position in this sequence.

Figure 1 represents a simple database containing documents related to the food service industry. The data was obtained through our local commercial contacts in the Waterloo area. The text is marked up using a variant of SGML[1]. Figure 2 shows the mapping of the documents into the

[1] Smilin' Guy Markup Language

database. Markup indicating interdocument boundaries ("☺") is indexed between words at positions $\frac{1}{2}$, $10\frac{1}{2}$, $17\frac{1}{2}$, $26\frac{1}{2}$, $37\frac{1}{2}$, and $45\frac{1}{2}$. We will use this database as an on-going example throughout the next several sections of the paper.

Given a query Q, an *extent* satisfying Q is a pair (p, q) of positions in the text such that the substring of the text database beginning at position p and ending at q satisfies the query. For simple boolean queries:

1. An extent (p, q) satisfies a query Q_1 `AND` Q_2 if the extent satisfies Q_1 and satisfies Q_2.

2. An extent (p, q) satisfies a query Q_1 `OR` Q_2 if the extent satisfies Q_1 or satisfies Q_2.

3. An extent (p, q) satisfies a term T if the term occurs in the interval of text represented by the extent.

From these definitions it is clear that a great many extents may satisfy a particular query. For example, if there is any extent that satisfies a query then the extent corresponding to the entire database also satisfies the query. For a query consisting of a single term, any extent in the database that overlaps an occurrence of the term satisfies the query. For the database of figure 2, the query

```
"you" AND "will"
```

is satisfied by the extents $(1, 12)$, $(2, 12)$, $(3, 12)$, $(12, 45)$ and $(34, 45)$ among others — there are hundreds — but not by $(1, 11)$, $(12, 17)$ or $(35, 45)$.

From the large number of extents that may satisfy a query, we take as solutions only those that have no other satisfying extents contained within them. That is, from a set of extents S satisfying a query we accept as solutions the set of extents $\mathcal{G}(S)$, where

$$\mathcal{G}(S) = \{(p, q) \mid (p, q) \in S \text{ and } \nexists\ (p', q') \in S \text{ such that } (p, q) \neq (p', q'),\ p \leq p' \text{ and } q \geq q'\}\,.$$

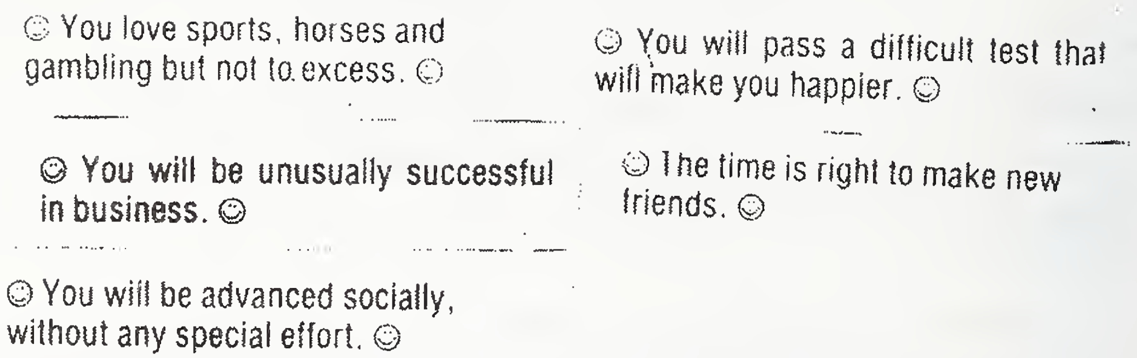
In the case of our example database the solutions are $(11, 12)$, $(12, 18)$, $(18, 19)$, $(19, 27)$, $(27, 28)$, and $(34, 36)$.

4 Ranking by Solution Density

When our work began for TREC-4 we had no experience with ranking solutions to GCL queries. Although adapting traditional ranking techniques to GCL was one possible route, we felt that the unique properties of GCL, particularly the shortest substring search model, might lead to the successful development of a more novel method. Although we are quite happy with the results of our TREC experiments, we wish to emphasize that the techniques we used, described in this section, are highly experimental and were invented within the timeframe of TREC-4, given impetus by the necessity that our participation created.

After some preliminary work we decided to base our ranking on two assumptions:

- *Assumption A*
  The smaller a solution extent, the more likely that the corresponding text is relevant.



☺ You love sports, horses and gambling but not to excess. ☺

☺ You will be unusually successful in business. ☺

☺ You will be advanced socially, without any special effort. ☺

☺ You will pass a difficult test that will make you happier. ☺

☺ The time is right to make new friends. ☺

Figure 1: Example Source Text

1	2	3	4	5	6	7	8	9
you	love	sports	horses	and	gambling	but	not	to
10	11	12	13	14	15	16	17	18
excess	you	will	be	unusually	successful	in	business	you
19	20	21	22	23	24	25	26	27
will	be	advanced	socially	without	any	special	effort	you
28	29	30	31	32	33	34	35	36
will	pass	a	difficult	test	that	will	make	you
37	38	39	40	41	42	43	44	45
happier	the	time	is	right	to	make	new	friends

Figure 2: Example Text Database

- *Assumption B*
  The more solution extents contained in a document, the more likely that the document is relevant.

The first assumption provides a ranking for individual solutions extents; the second suggests a ranking technique for particular documents in terms of solution extents. Both assumptions are superficially reasonable, and preliminary trials with the TREC data appeared to bear out the assumptions. However, a problem remained in combining these assumptions to produce a single score for a document.

Assumption B suggests that a document might be ranked by summing individual scores of solution extents contained within it. A natural value to use as the score of a particular extent (p, q) is its length $|(p, q)| = (q - p + 1)$. Unfortunately this approach assigns a higher score to less relevant documents. Summing individual score is reasonable only if a higher score indicates a more relevant document. To rectify the problem we considered an inverse relationship

$$\text{Score of } (p, q) \propto \frac{1}{|(p, q)|}$$

or

$$\text{Score of } (p, q) = S(p, q) = \frac{A}{|(p, q)|}$$

During our preliminary trials it was then quickly observed that if the length of an extent was below a threshold of a dozen or so words, assumption A no longer appeared to hold and all extents appeared equally relevant — and certainly not varying at the level indicated by an inverse relationship. Therefore, we took the score for a particular extent as:

$$S(p, q) = \begin{cases} \frac{A}{|(p,q)|} & \text{if } |(p, q)| \geq A \\ 1 & \text{if } |(p, q)| \leq A \end{cases}$$

For any extent (p, q), we have $0 < S(p, q) \leq 1$. For our Trec experiments, a value of 16 was used for the constant A.

For our example database, arbitrarily taking A to be 2, we get scores $S(11, 12) = 1$, $S(12, 18) = 0.29$, $S(18, 19) = 1$, $S(19, 27) = 0.22$, $S(27, 28) = 1$, and $S(34, 36) = 0.67$ for solution extents to the query

```
"you" AND "will"
```

If solution extents (p_1, q_1)...,(p_N, q_N) are contained in a particular document, the score for the document is

$$\sum_{i=0}^{N} S(p_i, q_i)$$

In determining the score for each fortune in figure 1 we take only the solution extents contained entirely in a single fortune

```
("you" AND "will") contained in ("☺" ...  "☺")
```

leaving extents $(11, 12)$, $(18, 19)$, $(27, 28)$ and $(34, 36)$. The score for the fourth fortune is highest $(S(27, 28) + S(34, 36) = 1 + 0.67 = 1.67)$; the second and third fortunes both receive a score of 1; the first and fifth fortunes receive a score of 0.

5 TREC Experiment

5.1 Procedure

The MultiText project participated in both the routing task and the ad-hoc task. Queries were developed manually, the procedure differed only slightly for the two tasks. The queries were created manually by two of the investigators (Clarke and Cormack) working in conjunction. Approximately 15 to 45 minutes was spent developing a query for each topic. During creation of the routing queries relevant documents were sometimes pulled and used as a source of possible terms, but this practice was not uniformly followed. Besides the personal knowledge of the investigators, the only external resources used were an on-line dictionary (Webster's); the Unix `spell` program; an on-line list of country, state and city names and state postal abbreviations; and, in some few cases, current issues of newspapers.

The final query developed for each topic was a compound query consisting of an ordered list of one or more sub-queries (2.05 sub-queries on average). Results for each sub-query were determined separately using the ranking techniques described in the previous section. These results were then combined into a final solution set according to the ordering of the sub-query list, with results of a particular sub-query ranked before the results of subsequent sub-queries. Documents given a non-zero score by one sub-query were eliminated from the results of subsequent sub-queries before this final ranking.

This approach reflects a trade-off between a desire for precision and an artificial need to produce 1000 ranked documents. The query appearing at the beginning of the list is intended to be a precise expression of the requirements underlying the topic. Queries occurring later in the list are "weaker" and are intended to pick up a large number of possibly relevant documents.

Figure 3 shows topic 246 and figure 4 gives our query in the internal format presented to the system and forwarded to NIST. In this format a terse syntax is used for operators, and phrases are expanded into term queries. Some explanation of the syntax is required: "`^`" and "`+`" are equivalent to "`AND`" and "`OR`" respectively, "`<>`" is equivalent to the ordering operator "`...`", the expression "`[2]`" is a query representing all two-word intervals in the text. The "`@output`" command sets the output file name for the query. The "`@rank`" command takes a topic number and a compound query as arguments and executes the ranking procedure. The topic number is used by the "`@rank`" command only for formatting the output. In this case the compound query list has only a single sub-query, which has been assigned the name "q".

The query of figure 4 is essentially a boolean expression in conjunctive normal form, consisting of three "facets", each built from several named pieces for convenience. The first facet ("`arms`") is a disjunction of terms and phrases related to military weapons. The second facet ("`export`") is a disjunction of terms related to trade. The final facet ("`USbroad`") is a disjunction of 150 geographical place names and abbreviations related to the United States. The definition of this last facet is not included in figure 4; its definition is global in scope and it is used whenever a topic concerns only the U.S.

Several other global definitions of this type were used in developing the queries and these definitions contributed significantly to the size of the queries. For the ad-hoc task, queries contained an average of 67 terms. For the routing task, queries contained an average of 53 terms. The variance was fairly high, for some topics the query consisted of hundreds of terms, for other topics the query

```
<top>

<num> Number:  246

<desc> Description:

What is the extent of U.S. arms exports?

</top>
```

Figure 3: Topic 246

```
@output "246.output"

arms0 = "arms" + "gun" + "guns" + "tanks"
arms1 = "firearm" + "firearms" + "weapon" + "weapons" + "rifle" + "rifles"
arms2 = (("fighter" <> ("jet" + "jets")) <  [2]) + "bomber" + "bombers"
arms = arms0 + arms1 + arms2

export0 = "export" + "exports" + "trade" + "sale" + "sales"
export1 = "tariff" + "tariffs"
export = export0 + export1

q = arms^export^USbroad

@rank 246 q
```

Figure 4: MultiText Query for Topic 246

algorithms. In *Overview of the Third Text REtrieval Conference (TREC-3)*, pages 179–199, 1994.

[3] Charles L. A. Clarke, G. V. Cormack, and F. J. Burkowski. An algebra for structured text search and a framework for its implementation. *The Computer Journal*, 38(1):43–56, 1995.

[4] Charles L. A. Clarke, Gordon V. Cormack, and Forbes J. Burkowski. Schema-independent retrieval from hetrogeneous structured text. In *Fourth Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, Nevada, April 1995.

[5] Heather Fawcett. *A Text Searching System — PAT 3.3 User's Guide*. Centre for the New Oxford English Dictionary, University of Waterloo, 1989.

[6] E. Fox, S. Betrabet, M. Koushik, and W. Lee. Extended boolean models. In William B. Frakes and Ricardo Baeza-Yates, editors, *Information Retrieval — Data Structures and Algorithms*, chapter 15, pages 393–418. Prentice Hall, Englewood Cliffs, NJ, 1992.

[7] D. K. Harman, editor. *Overview of the Third Text REtrieval Conference (TREC-3)*. National Institute of Standards and Technology, U. S. Department of Commerce, 1994. NIST Special Publication 500-225.

[8] David Hawking and Paul Thistlewaite. Searching for meaning with the help of a PADRE. In *Overview of the Third Text REtrieval Conference (TREC-3)*, pages 257–267, 1994.

[9] David Hawking and Paul Thistlewaite. Proximity operators — So near and yet so far. In *Fourth Text REtrieval Conference (TREC-4)*, 1995.

[10] The MultiText Project. More information may be found in the project repository: `ftp://plg.uwaterloo.ca/pub/mt`.

[11] Airi Salminen and Frank Wm. Tompa. Pat expressions — An algebra for text search. *Acta Linguistica Hungarica*, 41:277–306, 1994. A version of this paper is available as: Technical Report OED-92-02, UW Centre for the New Oxford English Dictionary, Waterloo, Ontario, Canada, 1992.

[12] Tadeusz Radecki, editor. Special issue on the potential for improvements in commerical document retrieval systems. *Information Processing and Management*, 24(3), 1988.

to the subject. In the previous TREC conference [7], Charoenkitkarn, Chignell and Golovchinsky [2] produced reasonable preformance using a simple boolean ranking technique along with highly interactive query development. For the same conference, Hawking and Thistlewaite [8] described experiments using the PADRE parallel free-text scanning system. The language used by the PADRE system is very similar to Pat. For ranking, they used a weighed sum based on term frequency and document length. In the current TREC conference, Hawking and Thistlewaite [9] continue their work with the PADRE system using a proximity scoring technique similar to ours.

7 Conclusions

For TREC-4, the MultiText project explored how the unique properties of our query language, GCL, could be exploited for document ranking purposes. Queries were primarily expressed in a boolean subset of the language, and the shortest substring property of the language was used as the basis for developing a scoring method. Based on our results, the techniques we developed appear to provide a simple, efficient and effective method for ranking GCL queries.

Since our ranking technique is relatively new, many aspects have not yet been explored. The scoring technique described in section 4 should be investigated in more depth. The sensitivity of ranking to changes in the parameter A is of interest. Other scoring formula could be developed. The query terms we used for the TREC experiments could be used with an entirely different ranking method to provide a more direct comparison than that provided by TREC, where the terms vary heavily from group to group.

For a future TREC conference we would concentrate our participation on the ad-hoc task and extend our participation to the interactive task. We are presently undertaking user interface and query construction research targeted toward the capabilities of GCL. This research should directly benefit our participation in an interactive task.

Acknowledgements

Rob Good provided system administration support and helpful feedback as the experimental work progressed. Bryan West, our Undergraduate Research Assistant, wrote a number of scripts to assist with data loading and result processing. We thank Sunshine Express for providing the example data used this paper.

The MultiText Project receives its primary funding from the Government of the Province of Ontario through its Information Technology Research Centre. Additional funding was provided by the Natural Sciences and Engineering Research Council of Canada.

References

[1] Forbes J. Burkowski. An algebra for hierarchically organized text-dominated databases. *Information Processing and Management*, 28(3):333–348, 1992.

[2] N. Charoenkitkarn, M. Chignell, and G. Golovchinsky. Interactive exploration as a formal text retrieval method: How well can interactivity compensate for unsophisticated retrieval

consisted of a single two-term phrase. Overall, about half of the query terms resulted from the expansion of global definitions, overwhelmingly from the expansion of the "USbroad" definition. Expansion of phrases into terms and the manual construction of morphological term variants also contributed to the large number of terms per query.

5.2 Results and Discussion

Our results for the ad-hoc task are quite reasonable (average precision: 0.2994; R-precision: 0.3347). For over 65% of the topics our average precision is above the median average precision for all groups. Our results for the routing task are relatively poor (average precision: 0.1188; R-precision: 0.1649). In most cases our average precision is below the median average precision.

Given the similarity in methodology these results are surprising. After attempting to explain the difference, we discovered that the Ziff data from disk 3 had been omitted inadvertently from the routing run. In our final results, no Ziff documents were reported, and the overall recall results are commensurately lower.

The ranking technique is reasonably efficient. For the ad-hoc task, system search time required an average of 40 seconds per query (an average of 18 seconds per sub-query). For the routing task, system search time required an average of 10 seconds per query (an average of 5 seconds per sub-query). Nonetheless, the system is a research prototype and performance tuning and known algorithmic enhancements should reduce these numbers significantly. Since the results were submitted, some simple performance tuning, requiring less than an evening's work, have made the runs approximately 40% faster.

The scoring procedure described in section 4 produces scores that are independent of the characteristics of other documents in the collection — inverted document frequency, for example. This property allows a collection to be split arbitrarily into a number of sub-collections to be searched in parallel by separate search engines, with the results merged for the final ranking. This approach would produce essentially a linear speed-up in search time with the number of engines used.

6 Related Work

GCL owes some of its intellectual and cultural heritage to two earlier structured-text retrieval languages developed at the University of Waterloo. The language of Burkowski [1] is the direct ancestor of GCL, but primarily focused on structural queries over pre-defined hierarchical document components. The capabilities of that language are extended substantially by GCL.

The Pat text searching system [5, 11] was developed for use with the New Oxford English Dictionary. Queries expressible in Pat (with a few exceptions) are a subset of those expressible in GCL. Boolean queries in Pat must be solved with respect to a particular document component or fixed proximity, making the results of this paper inapplicable. Semantic limitations in Pat make the solving of certain types of structural queries, such as the earlier examples involving pairs of paragraphs, impossible.

Because of the prevalence of boolean queries in commercial text retrieval systems, the ranking of boolean queries has been the subject of extensive research. Fox et al. [6] provides an overview of several methods; a special issue of *Information Processing and Management* [12] has been devoted

CLARIT TREC-4 Experiments

David A. Evans[1,2], Nataša Milić-Frayling[1], Robert G. Lefferts[1]

[1]CLARITECH Corporation
319 South Craig St., Suite 200
Pittsburgh, Pennsylvania 15213-3726

[2]Laboratory for Computational Linguistics
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213-3890

1 Introduction

The CLARIT TREC-4 research effort has focused principally on aspects of CLARIT processing that can be influenced by the user in an otherwise automatic retrieval process. This paper describes the preparation and results of the CLARIT system in the *manual ad-hoc retrieval* task and a companion paper in this volume (Milić-Frayling et al. 1996) describes the CLARIT experiments in the *interactive retrieval* task. To provide a context for the discussion of our work in TREC 4, we offer, first, a review of the evolution of the CLARIT system in past TRECs.

2 A Brief History of CLARIT in TREC

As illustrated in Table 1, the CLARIT system offers many alternative choices in document and query processing, in scoring, and in providing information to supplement the retrieval process. For example, native CLARIT processing supports indexing by terms that are based on linguistic constituents (phrases, sub-phrases, words), where elements can be morphologically 'normalized' or not; derived index terms may be 'typed' or not (e.g., as noun phrases, or verbs, or modifiers, etc.); texts may be indexed as whole objects or on the basis of sub-documents whose size can vary; index terms can be weighted in a number of ways; query terms can also be weighted and modified by coefficients that can be positive, negative, or zero; supplemental terms can be derived from a number of sources, including the whole target corpus or a sample set of documents or an automatically generated first-order thesaurus (which, in turn, can be created in a variety of different ways); the 'score' of a document to a query can be based on vector-space measures or 'feature counts'. As summarized in Table 2, the history of CLARIT in TREC is a history of the exploration of parameters such as these.

In addition, as a design philosophy, CLARIT processing (and hence the design of CLARIT–TREC experiments) has emphasized minimal (and simple) preprocessing of queries or corpora, the use of only those supplemental resources that can be generated 'on the fly', ease of user interaction, and speed.

- NL Processing of Queries and Documents
  - Choice of NL constituents
    —NPs, Vs, Mods, etc.
  - Processing of texts as whole docs
    or collections of (possibly overlapping) sub-docs
  - Sub-doc sizes
  - Term space representations
    - NPs with single word subterms
    - NPs with attested subterms
    - Only multi-word
      or only single-word terms
    - Combinations of nouns (and NPs),
      Adjectives, and Verbs, etc.
- Query Term Weighting
  - Default query-term weights (1–1–1 or 3–2–1)
  - Positive vs. "negative" terms
  - Negatives with coefficients of "0" or "–N"
  - Weighting based on corpus (or general) statistics
- Term-Space Expansions
  - Augmented terms from thesaurus
  - "Distractor" terms
  - Sample of corpus terminology
  - Using terms from individual query
    vs. joined queries
- Thesaurus Terms
  - All terms vs. above-"threshold" terms
  - Selections based on sub-docs
- "Feedback" Documents
  - All top-N
    vs. thesholded top
    vs. filtered top
- Scoring Function
  - Feature scoring vs. cosine distance
  - IDF–TF vs. other
  - Doc score based on max. sub-doc score
    vs. whole doc
    vs. combination

Table 1: A Subset of CLARIT Parameters

CLARIT in TREC			
Conference	Goals/Focus	Approach/Techniques	Observations/Outcomes
Pre-TREC	• Minimal Preprocessing • On-the-Fly Resource Discovery • Simple Queries • Adapting to User • Speed, Efficiency	• Natural-Language Processing (NLP) • Vector-Space Matching • Thesaurus Discovery • Term Weighting	
TREC-1	• General Architecture • Scaling to TREC • High Precision • Rapid Query Formulation	• Indexing/Scoring on Full Documents • Query Term Coefficients • Term/Sub-Term Weighting • Relevance Feedback with Thesaurus Discovery • Partioning on Working Set / Indexing / Scoring	• Utility of NLP • General Applicability of Thesaurus Discovery • Value in Query Expansion • Need for Coherent Architecture
TREC-2	• Streamlining • Practical Document Normalization • Fully-Automatic Processing	• 'Squery'—One-Shot Querying of DB / No Inversion • Processing on Sub-Documents • Automatic Feedback / Query Expansion with Thesaurus Discovery • Scoring on Best Sub-Document • Minimal Discourse Processing of Query	• Power of Thesaurus Discovery and Query Expansion • Need to Understand Parameters in the Process • Need to Control Term Space
TREC-3	• Parameterizing the Process • Calibrating Term Weighting • Exploring Term-Space Variants • Revising Document Scoring	• Distractor Terms (Positive/Negative Feedback) • Independent Term Space • Scoring on Full Documents • Control on Thesaurus Sub-Document Size	• Very Little Time! • Positive Effect of Using Independent Query Term Space • Need for Better Control on Scoring/Sub-Document Selection • Need for Control in Selecting Positive/Negative Terms for Query Expansion
TREC-4	• 'User' Support • Automatic, Variable Control on Feedback • Revisiting Document Scoring • Stabilizing Term-Space Effects	• 'Attesteds' in Document/Query • 'Required Terms' Filter in Feedback • Mining External DBs to Assist Query Formulation • 'Global'/'Local' Perspective in Document Scoring • Overlapping Sub-Documents • Positive/Negative Terms	• Micro Time!! • Filtering Needs Work • Simple User Interventions can have Dramatic Effects on System Performance • More Experiments are Needed

Table 2: An Overview of CLARIT in TREC

2.1 CLARIT–TREC-1—Initial System Configuration

The main goal for CLARIT in TREC-1 was to develop an architecture for performing TREC evaluations and to re-scale CLARIT tools to handle databases of TREC size. In addition, we aimed for high precision (especially in the initial step of automatic processing) and rapid (simple) query formulation. While the configuration for the CLARIT–TREC system has been almost completely revised since the TREC-1 effort, the system continues to employ the same core techniques:

1. NLP applied to all documents and topics;
2. thesaurus extraction, for query enhancement; and
3. vector-space modeling for relevance ranking.

The use of such techniques in the system has undergone extensive refinement and modification, but they continue to play central roles in current versions of the CLARIT system.

The special additional techniques we used in CLARIT–TREC-1 processing included using full documents (not sub-documents) for scoring, weighting query terms with simple (3–2–1) coefficients, to assign importance to terms, indexing on noun phrases as well as their single-word constituents, retrieving a working set of documents (to partition the corpus) and then refining the search within the working set, and using thesaurus discovery to identify terms from potentially relevant documents to supplement the source query for a second-pass retrieval. (Cf. Evans et al. 1993 for more detail.)

In general, we felt our TREC-1 goals were achieved. We noted especially the value of NLP in identifying useful phrases (enhancing precision) and the power of CLARIT thesaurus discovery to find supplemental terms for query expansion. We also saw the need for a simpler processing architecture to accommodate future TREC tasks.

2.2 CLARIT–TREC-2—Towards Full Automation

While the CLARIT–TREC-1 system demonstrated the efficacy of core CLARIT technologies, deeper evaluation of individual techniques was possible only after the technical advancements of the TREC-2 system. The CLARIT–TREC-2 system reflected a streamlined architecture for fully automatic processing, supporting parametrization and analysis. In particular, the TREC-2 system was capable of

1. discovering potentially useful documents for automatic feedback;
2. identifying portions of the documents (sub-documents) that would be the best source of additional terminology; and
3. extracting characteristic terms and recalibrating the initial query automatically.

The usefulness of CLARIT automatic feedback was demonstrated by comparing CLARIT–TREC-2 retrieval results against a baseline without such feedback. As summarized in Table 3, the TREC-2 manual routing task showed a 69% relative improvement in average precision, the automatic routing task 76%. Such dramatic query augmentation results are accounted for, in part, by the use of reliable sources of relevant documents (associated with routing topics) in the CLARIT thesaurus discovery process. However, significant improvement in average precision can be seen even in the absence of relevance judgments, as was demonstrated in the TREC-2 ad-hoc experiments. For manual ad-hoc queries improvement was 21% and for automatic queries about 22%. The results were especially interesting since the documents used for feedback were automatically identified by the system in an initial round of text retrieval. We believe that the effectiveness of the automatic process was due to the precision of the initial retrieval results over the target corpus, although the process might also have been affected by certain properties of the corpus itself, for example, the relatively large number of relevant documents for each TREC topic. Other groups at TREC have experimented with term expansion to improve performance, noticeably those at UCLA (Efthimiadis & Biron 1994) and Cornell (Buckley et al. 1995).

Other techniques that were important in CLARIT–TREC-2 processing included the use of sub-documents (not whole documents) to produce a document similarity score; the first use of simple discourse heuristics to process queries automatically; and an approach to batch processing the TREC corpus against queries without indexing the corpus. (Cf. Evans & Lefferts 1994, 1995.)

TREC-2 results were very encouraging and established a baseline for use in evaluating future work. The power of automatic query expansion was clearly demonstrated. Nevertheless, subsequent experiments in which even better results were obtained via minor adjustments to the thesaurus discovery and term expansion process underscored our need to understand the interaction of the many parameters in the system.

CLARIT TREC-2	Rel.Improvement in Avg.Prec.	
Experiments	Automatic	Manual
Routing	76%	69%
Ad-Hoc	22%	21%

Table 3: Summary of Results from Automatic Query Expansion in the CLARIT–TREC-2 System

2.3 CLARIT–TREC-3—Explorations of Term Space

We had several goals in TREC 3, including a finer calibration of term weighting, document scoring, automatic feedback mechanisms, and, most importantly, term-space representations. The size and structure of term spaces, in particular, seemed to affect CLARIT processing significantly.

In CLARIT processing, a *term space* is simply the list of terms that are used as index features for a given document or query. As illustrated in Figure 1, how the term space is chosen can have consequences for a process such as vector-space relevance (or similarity) scoring.

In traditional vector-space models, the term space is a static set: the (unique) union of all terms found in all documents in the corpus. (Terms can be nominated by whatever term-discovery mechanism is available: identification of 'words', with or without stemming; NLP for phrases and sub-phrases; etc.) If a query contains terms that are not found in the term space, they are ignored, since they do not match terms in any documents. Thus, the presence of such terms in the query will not contribute to calculation of the similarity score of the query to any document in the corpus, for example, via the use of a formula such as the one given in Equation 1.

An alternative term space can be determined by virtually any selection of terms. For example, all the terms in a reference work, such as a medical dictionary, could be used to establish a term space. If such a space were chosen for the processing of queries against a corpus, only the terms in the query and corpus that matched 'medical terminology' would be used in determining the similarity or fit of a query to a document. Queries about medical topics would score well against documents with descriptions of medical phenomena; other queries and documents would score very poorly or not at all. The choice of terms, therefore, can be used to effect a bias or 'perspective' on the matching process.

In the batch process/no-prior-indexing approach taken with the CLARIT–TREC-2 system, we sought to limit the term space to increase general processing speed by reducing the number of features in any document that we needed to score. Since the contrasts we wanted to maximize were just those discriminating one query from another, we decided to use the unique union of the terms in the TREC-2 queries as the term space for the process. The fifty routing topics were run in parallel in one unified term space, and the fifty ad-hoc topics were evaluated in their own unified term space. Consequently, when scoring a document against a particular query (using a cosine distance measure), the terms specific to the query provided positive evidence for the relevance of a document. All other terms in a document that were 'activated' by the unified term space provided negative evidence by increasing the document length and therefore reducing the document similarity score. Since a decrease in document score is directly related to the sharing of terminology with other queries, the retrieval model provided a natural query discriminating mechanism.

We subsequently repeated the TREC-2 experiments with a full term space (based on the terms in the corpus) and got significantly poorer results. This suggested to us that limited term spaces, closely tied to the queries, not the corpus, can be effective in identifying relevant documents. But we recognized that the particular term space we used for TREC-2 tasks was an artifact of the TREC process (involving fifty topics) and not the basis for an independent calibration of a query to a corpus. Thus, in TREC 3, we focused especially on techniques for determining—on a query-by-query basis—an 'independent' term space for each topic. One result was the creation of a new vector-space model for the CLARIT–TREC-3 system, one in which the term space is dynamic and the vector representation of any document can be constrained (or tuned to a query) on the fly. In this way, the term space provides a unique evaluation context for any given query.

In our TREC-3 experiments, we explored different mechanisms for specifying the term-space:

1. a global term space, in the fashion of traditional vector-space models;
2. a unified-query term-space, where the term space is just the unique union of all of the terms found in all topics being processed; and
3. an independent-query term-space, where the term space for a query is just the vector of terms specified in the query itself.

Some of the properties of different approaches to term-space setting are illustrated in Figure 1. In particular, the effect on similarity score is highlighted in the 'toy' example. Note the impact that a change in term-space context has on query–document similarity scores. As we observed above, terms that are not present in the term space are ignored in the query and they disappear from the inner product in the numerator. However, with non-traditional approaches, the document length factor in the denominator is no longer fixed for a given document; it depends on the terms present in the document vector as constrained by the term space.

Therefore, the CLARIT–TREC-3 effort involved the implementation of a retrieval procedure with individual-query term spaces and the exploration of the interaction between term-space parameters and other processes in the system, including automatic query augmentation and final document scoring.

General Formula for Measuring Similarity in a Term Space:
For any query Q_i and document D_j, their similarity is given via an evaluation of shared and disjoint features in an orthogonal space of t terms, typically as shown below in Equation 1.

$$S(Q_i, D_j) = \frac{Q_i \cdot D_j}{|Q| \cdot |D|} = \frac{\sum_{k=1}^{t}(q_{i_k} \cdot d_{i_k})}{\sqrt{\sum_{k=1}^{t} q_{i_k}^2} \cdot \sqrt{\sum_{k=1}^{t} d_{i_k}^2}} \tag{1}$$

A Sample Set of Queries and Documents:

Terms	Q₁	Q₂	D₁	D₂
dog	1	1	2	—
cat	1	—	—	1
hat	1	—	1	—
bat	1	—	—	—
mat	1	1	—	—
hut	—	1	2	—
cut	—	1	—	2
luck	—	—	3	—
buck	—	—	1	—
muck	—	—	—	3

Global Term Space:
The term space as determined by the unique union of D_1 and D_2 is $\{dog, cat, hat, cut, hut, luck, buck, muck\}$. In *global term space,* the similarity measure between D_1, whose active terms are $\{dog, hat, hut, luck, buck\}$, and Q_1 and Q_2, respectively, is as given below:

$$S_G(Q_1, D_1) = \frac{(1 \cdot 2) + (1 \cdot 1)}{\sqrt{1^2 + 1^2 + 1^2 + 1^2 + 1^2} \cdot \sqrt{2^2 + 1^2 + 2^2 + 3^2 + 1^2}} = \frac{3}{\sqrt{5} \cdot \sqrt{19}} = 0.31 \tag{2}$$

$$S_G(Q_2, D_1) = \frac{(1 \cdot 2) + (1 \cdot 2)}{\sqrt{1^2 + 1^2 + 1^2 + 1^2} \cdot \sqrt{2^2 + 1^2 + 2^2 + 3^2 + 1^2}} = \frac{4}{\sqrt{4} \cdot \sqrt{19}} = 0.46 \tag{3}$$

Unified Query Term Space:
The term space as determined by the unique union of Q_1 and Q_2 is $\{dog, cat, hat, bat, mat\ hut, cut\}$. In *unified query term space,* the similarity measure between D_1, whose active terms are $\{dog, hat, hut\}$, and Q_1 and Q_2, respectively, is as given below:

$$S_U(Q_1, D_1) = \frac{(1 \cdot 2) + (1 \cdot 1)}{\sqrt{1^2 + 1^2 + 1^2 + 1^2 + 1^2} \cdot \sqrt{2^2 + 1^2 + 2^2}} = \frac{3}{\sqrt{5} \cdot \sqrt{9}} = \frac{1}{\sqrt{5}} = 0.45 \tag{4}$$

$$S_U(Q_2, D_1) = \frac{(1 \cdot 2) + (1 \cdot 2)}{\sqrt{1^2 + 1^2 + 1^2 + 1^2} \cdot \sqrt{2^2 + 1^2 + 2^2}} = \frac{4}{\sqrt{4} \cdot \sqrt{9}} = \frac{4}{\sqrt{6}} = 0.67 \tag{5}$$

Independent Query Term Space:
The term space as determined by Q_1 is $\{dog, cat, hat, bat, mat\}$ and by Q_2 is $\{dog, mat, hut, cut\}$. In *independent query term space,* the similarity measure between D_1, whose active terms are $\{dog, hat\}$, and Q_1 and Q_2, respectively, is as given below:

$$S_{Q_1}(Q_1, D_1) = \frac{(1 \cdot 2) + (1 \cdot 1)}{\sqrt{1^2 + 1^2 + 1^2 + 1^2 + 1^2} \cdot \sqrt{2^2 + 1^2}} = \frac{3}{\sqrt{5} \cdot \sqrt{5}} = \frac{3}{5} = 0.60 \tag{6}$$

$$S_{Q_2}(Q_2, D_1) = \frac{(1 \cdot 2) + (1 \cdot 2)}{\sqrt{1^2 + 1^2 + 1^2 + 1^2} \cdot \sqrt{2^2 + 2^2}} = \frac{4}{\sqrt{4} \cdot \sqrt{8}} = \frac{1}{\sqrt{2}} = 0.71 \tag{7}$$

Figure 1: Examples Illustrating of the Effects of Term-Space Differences on Query–Document Similarity Scores

Queries in the CLARIT–TREC-3 system are composed of layers of terminology that specify both the query vector and the query's term space. Terms from the initial topic description and terms extracted from training or feedback processing are used as positive evidence for document relevance. In addition, each query expresses a query-specific distractor space: a set of terms that help differentiate between truly relevant and false-positive ('distracting') documents. Such terms are extracted either from actual false-positive documents (in the case of routing queries where known relevants are given) or from a sample of assumed non-relevant documents determined by sampling or intial retrieval results (in the case of the ad-hoc task).

Although the purpose and the discrimination value of the distractor space is quite different from that in a unified query term space, some of the effects of query vs. query sensitivity are captured. The distractor terms for the routing experiment are extracted from documents that are falsely retrieved from the global training corpus by the initial query; they therefore represent terms that might be responsible for the confusion of topics during the routing of new material. For ad-hoc topics, the distractor space is constructed from the terms in the target corpus; these represent effectively a sampling of terms that appear in related but probably not relevant documents. In particular, the system selects terms from documents that are ranked low by the initial query. The documents therefore do contain some query terms, though most likely with a different sense than or in contexts inappropriate to the terms in the source query. The selection of such distractor terms is 'safe' since the procedure for incorporating discovered terms into the query vector defaults to terms that are already present in the vector: if a term already appears in the positive-term query vector, it will not be replaced by a matching term found in the distractor space.

The design of the individual-query term space thus supports both the group discrimination required with routing topics and the idiosyncratic sense and context discrimination required for ad-hoc queries. Our experimental results in preparing for TREC 3 indicated that independent-query term spaces yield a significant performance improvement over both global and unified-query term-space approaches, but the strong results we saw in preliminary experiments were not reflected in our official CLARIT–TREC-3 submissions. We cannot fully account for the relatively disappointing performance of the CLARIT–TREC-3 system, though we suspect the principal source of the problem derives from our failure to calibrate all the interactions among parameters before launching final runs. In addition, due to a variety of circumstances, the CLARIT team had only five days to devote to TREC-3 tasks.

3 CLARIT–TREC-4—Focus on the User

CLARIT–TREC-4 experiments had several goals. First, we sought to stabilize the independent-query term-space effects we saw in TREC-3 experiments. Second, we aimed to develop and test a mechanism for filtering the sub-documents chosen for use in finding positive-term candidates for query expansion. Third, we wanted to revisit the question of how best to normalize document scores. Finally, we wished to open a new area of exploration directed at the effective use of the user's input to maximize the benefits of automatic processes such as query refinement and augmentation and document scoring and ranking.

The user-focused tracks in TREC 4 provided a perfect opportunity to study a range of possible user interactions with the system and their effects on CLARIT automatic processes. Since we chose to concentrate on an exploration of ways to assist the user in creating initial queries that would be most effective in subsequent automatic processing, we decided to participate only in the manual ad-hoc task and the new interactive task.

In the balance of this paper, we report on the configuration, preliminary experiments, processing, and results of the CLARIT–TREC-4 system on two manual ad-hoc tasks, CLARTN and CLARTF. We also describe the results of post-TREC experiments that were designed to assess further the effects of user intervention on system performance.

3.1 Baseline Configuration

The CLARIT–TREC-4 ad-hoc experiments, CLARTF and CLARTN, are two in a sequence of experiments on user-system interaction that we intend to perform and analyze. CLARTN represents the latest baseline configuration of the CLARIT–TREC system, incorporating all of the techniques developed in the previous TRECs along with several new mechanisms. CLARTF uses the same configuration as CLARTN, but uses a manually constructed required-terms filter to discriminate among document windows used for feedback. The new features introduced into the CLARIT–TREC-4 baseline system include:

- *Document and query-term decomposition into attested sub-terms*

  Since CLARIT NLP produces full noun phrases as candidate index terms, and since such terms are typically multi-word phrases, some method for allowing partial matches between similar terms is required. In previous CLARIT–TREC systems we have used the single words from full terms as independent indexing features for the purposes of computing the similarity score. In TREC-4, we used a set of sub-terms consisting of all contiguous sequences of words from a term that can be identified as having occurred independently in the corpus. In other words, any sub-term that is *attested* independently in the corpus is regarded as a meaningful atom of any longer term that includes it and is used as an index to the full term, effecting another form of partial term matching.

- *Document indexing based on overlapping windows*

  The CLARIT–TREC-2 and CLARIT–TREC-3 systems used roughly paragraph-sized units as sub-documents in the processing of documents for thesaurus discovery (in automatic feedback) and, in the case of TREC 2, in scoring documents for relevance ranking. The CLARIT–TREC-4 system used fixed-length, overlapping document windows, similar to those used by the UMass group (Callan & Croft 1994).

- *Document scoring using a weighted average of the score of the best matching document window and the score of the full document*

  In addition to adopting fixed-length overlapping document windows, we have explored alternative scoring techniques that take advantage of global and local perspectives of document relevance. By combining the score of the document as a single vector with the score for the best scoring document window in the document, we are able to achieve substantial performance improvements.

- *Controlled feedback by means of a required terms filter*

  In order to refine the effects of automatic feedback, we have introduced a required terms filter that constrains the documents selected for positive feedback. A set of terms is manually specified for each query, and a document window is required to contain the terms (or some combination of them) before it will be considered useful for feedback. The filter is actually specified as a set of regular expressions in conjunctive normal form; a given document window passes the filter if at least one regular expression in each of the conjuncts matches some term in the window. Such a filter has value only for ad-hoc tasks, since routing tasks already include sample known-relevant documents.

The contributions of some of the new techniques in the CLARIT–TREC-4 system are evident in the results of a series of preliminary experiments we conducted preparatory to final system configuration.

3.2 Preliminary Experiments

In preparing for the TREC-4 task, we performed several preliminary experiments to calibrate the new components of the TREC-4 architecture. The experiments used the CLARIT–TREC-3 system on the TREC-3 ad-hoc task as a baseline against which to compare candidate new configurations. Note that we did not attempt to evaluate the effectiveness of the required terms filter in the preliminary experiments, but rather chose to use the official TREC-4 runs for that purpose. However, all of preliminary experiments did use the required terms filter by default.

3.2.1 Document Windows

A standing problem for the scoring of documents against a query is the determination of relevance. Is a long document with a small but concentrated set of matching features more or less valuable than a short document with same features distributed more uniformly throughout? If one processes all documents as collections of sub-documents and uses the score of the maximum scoring sub-document as the document score, then one typically elevates the relevance judgment of the longer document with concentrated features. While no fully automatic assessment of relevance is perfectable, we believe that a combination of perspectives on document relevance should be more effective than any one perspective alone. Our first experiment attempted to assess this hypothesis.

We reprocessed the TREC-3 ad-hoc corpus using whole documents and also using overlapping fixed-size "windows" to score similarity. We then rescored documents using a weighted average of the global and window (or local) scores. For a weighted average final score (FS) of a document (D_j) against a query (Q_i), we combine the score of the the maximum-scoring sub-document or window ($\hat{W}_{D_j}$) with that of the whole document generally as follows:

$$FS(Q_i, D_j) \quad = \quad \frac{\alpha \cdot S(Q_i, D_j) + \beta \cdot S(Q_i, \hat{W}_{D_j})}{\alpha + \beta} \qquad (8)$$

α and β are coefficients that can bias the perspective.

Figure 2 gives the results of three approaches to scoring. For the weighted average, we used overlapping sub-documents (windows) of fixed size (fifty terms) and $\alpha = 20$ and $\beta = 1$, giving the scoring formula as follows:

$$FS(Q_i, D_j) \quad = \quad \frac{20 \cdot S(Q_i, D_j) + S(Q_i, \hat{W}_{D_j})}{21} \qquad (9)$$

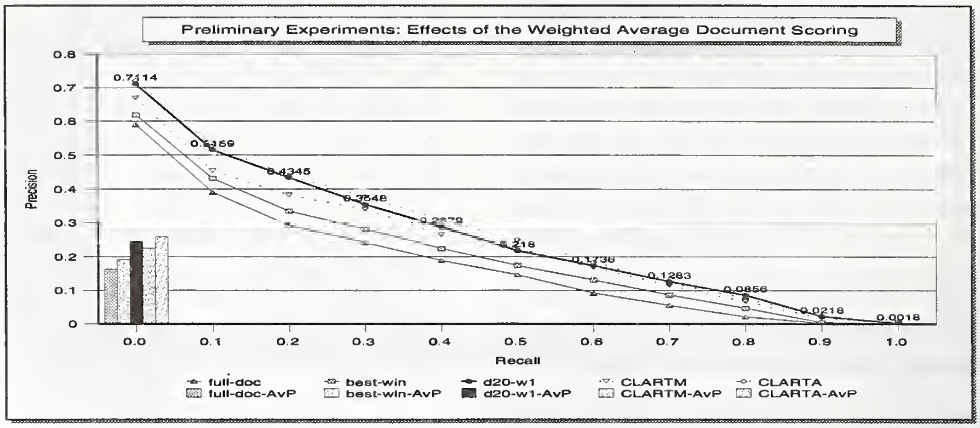


Figure 2: Comparative Scoring Effects: Full-Documents vs. "Best Window"vs. Weighted Combination

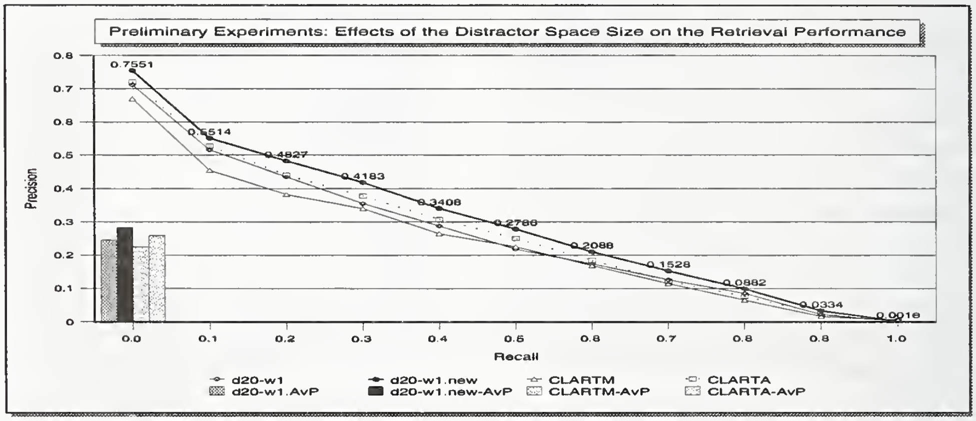


Figure 3: Comparative Scoring Effects: Distractor Space Size

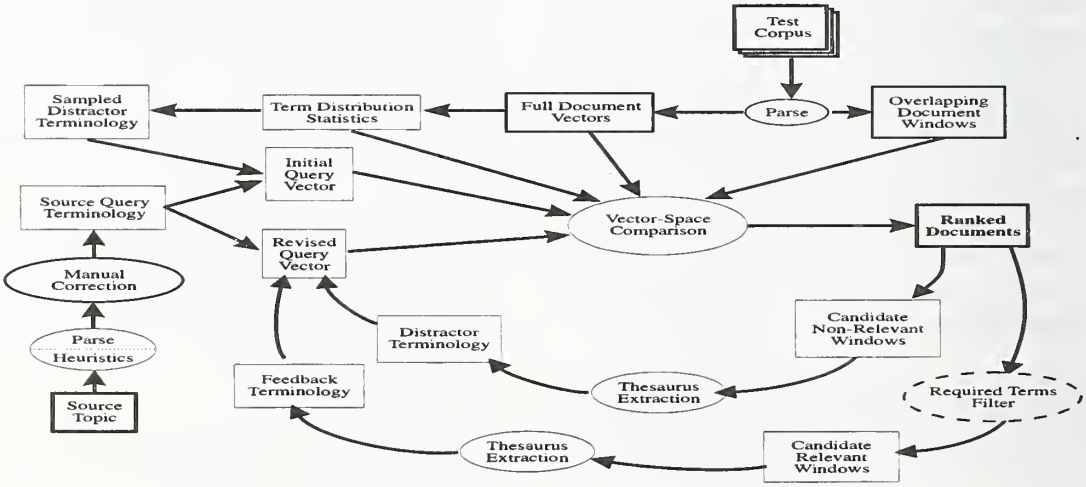


Figure 4: Schematic Representation of CLARIT–TREC-4 System Processing

As shorthand, we refer to this formula as "d20-w1". It is evident that d20-w1 performs better than either the global perspective of the whole document ("full-doc") or the local perspective of the best document window ("best-win"). However, weighted average scoring still performs at approximately the same level as was achieved with the CLARIT–TREC-3 system, shown as "CLARTM" (manual) and "CLARTA" (automatic).

3.2.2 Size of Distractor Terms Space

In a second series of experiments, we determined that the use of overlapping document windows (replacing non-overlapping sub-documents) had adversely affected the size and structure of the individual-query distractor spaces. We are able to overcome the effect by increasing the number of windows used to generate the distractor space and by reducing the thesaurus extraction threshold in the term-discovery process.

Figure 3 illustrates the effect of various sizes of the distractor space. The test run labeled "d20-w1" used the CLARIT–TREC-3 default distractor space configuration of 1,000 document windows and a 75% extraction threshold, while the run labeled "d20-w1.new" used a distractor space extracted from 1,500 documents with a 70% threshold. The results not only verify the effectiveness of the weighted average scoring technique, but also underscore the complicated dependencies between system parameters. As can be seen in the graph, the d20-w1.new run was significantly better than the best CLARIT–TREC-3 results.

3.3 Description of the Experiments: CLARTN and CLARTF

As noted above, the basic architecture of the CLARIT–TREC system has remained fairly constant since TREC 2. The system generates an initial query vector from the topic description and, for manual tasks, the user is allowed to modify and refine it. To identify document windows for automatic feedback, the initial query vector is submitted to an initial round of retrieval; the results are used as the input to the thesaurus extraction process, generating additional terminology. The terms nominated by the automatic feedback loop are added to the query vector and the reformulated query is then resubmitted to vector-space retrieval for final document ranking.

The CLARIT–TREC-4 process, given schematically in Figure 4, includes several additional processing steps that have been introduced since TREC 2. First, the initial query vector is augmented with a "sampled distractor space" that is generated from the test corpus. Second, the test corpus is analyzed both in terms of full documents and in terms of overlapping document windows. Document windows are output during the initial round of querying for automatic feedback, while the scores for both perspectives are averaged for final querying. Third, the results of the initial querying are used for positive feedback and to generate the distractor space, as described above. Finally, in the CLARTF run the required terms filter was applied to constrain the selection of documents used for automatic feedback. The CLARTN run, which did not use the filter, serves as a baseline against which to measure filtering effectiveness.

Another way to understand the CLARIT–TREC-4 system architecture is to consider the layers of terminology that are used to construct a query vector. (Cf. Figures 5 and 6.) A complete query representation contains the following classes of terminology:

1. *Query source terms that are extracted from the topic description by means of CLARIT natural language processing*
   In TREC 4, all query source terms were subsequently processed manually to modify terms, adjust term importance coefficients, delete noise terms, and add additional search terms as appropriate.

2. *Sampled distractor terms that are added to the initial query vector*
   These are simply the 2,000 most common single-word terms in the database. Sampled distractor terms are removed from the final formulation of the query vector and replaced by negative feedback terms that are specific to each topic.

3. *Positive feedback terms that are extracted from the candidate relevant document windows by CLARIT thesaurus extraction*
   The highest scoring 50 text windows are automatically submitted to thesaurus extraction using a 60% extraction threshold. If the required terms filter is applied, then the highest scoring 100 documents are tested against the filter, and the first 50 that pass the matching constraint are submitted to thesaurus extraction.

4. *Negative feedback terms that are also extracted from assumed non-relevant document windows retrieved by the initial query vector*
   However, the windows used to generate negative feedback terms are assumed to be not relevant to the query. A CLARIT thesaurus is extracted from 1,500 document windows that do not score well against the initial query. In fact, windows that are ranked between 37,500 and 39,000 are used for each query. The thesaurus is extracted with a threshold of 70%.

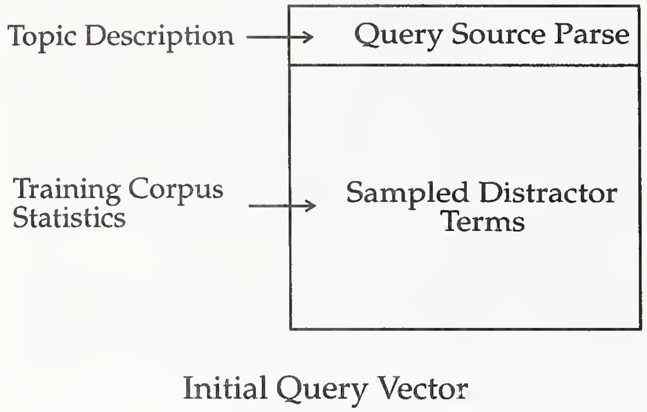


Figure 5: Composition of the Initial Query Vector

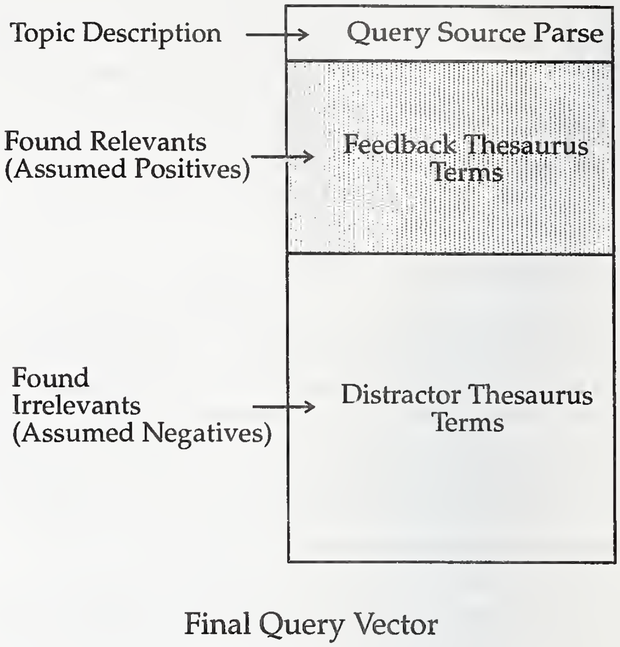

Topic Description
Query Source Parse
Found Relevants (Assumed Positives)
Feedback Thesaurus Terms
Found Irrelevants (Assumed Negatives)
Distractor Thesaurus Terms
Final Query Vector

Figure 6: Composition of the Final Query Vector

3.4 CLARIT–TREC-4 Processing

3.4.1 Initial Query Formulation

In both experiments, CLARTN and CLARTF, the initial query was prepared manually. The terminology that was obtained by automatic natural language processing of a topic description was reviewed, modified, and enhanced in interaction with the CLARIT IR system. TREC-4 ad-hoc topics were relatively short compared to those in previous TREC ad-hoc experiments, so we decided to use Disk 1 data to identify additional terminology that might be useful in the search. The user (not a CLARIT expert) performed interactive searches over the Disk-1 data using the CLARIT IR system with its standard user interface and produced thesauri to suggest candidate supplemental terms. In practice, such 'mining' of non-target-corpus databases consumed only a few minutes of user time.

The two main activities in tuning query formulations thus involved (1) selection of terminology to be added to the query and (2) adjustment of the importance coefficients of the query terms.

Initial query terms can be classified roughly into two categories. The first category consists of terms whose presence in the text provides positive evidence of document relevance. Such terms were assigned coefficients of 1 or greater. Discovery of additional 'positive' terms was facilitated by thesaurus extraction: documents selected by the user were processed to identify characterizing terminology and the user decided which of the terms to add to the query. The second category of terms consists of those that are responsible for retrieval of false-positive documents. Such terms were assigned 0 or a negative coefficient. 'Negative' terms were nominated by the user based on his or her familiarity with the topic and the system, possibly based as well on terms found in documents (from non-target databases) judged to be irrelevant by the user.

In addition to modification and enhancement of queries, users were asked to identify terms that might prove useful in filtering documents for feedback. Filter terms were selected as terms whose presence in a document would be a good indicator that the document (or a window containing the terms) would likely be relevant.

3.4.2 Database Preparation

Two new features used in document preparation were (1) partitioning of documents into overlapping windows and (2) indexing using attested subphrases. The window in both CLARTN and CLARTF experiments was 50 terms (noun phrases).

3.4.3 Initial Search

Manually prepared queries were used in the initial search over the TREC-4 ad-hoc database to identify documents that could be used for automatic query expansion. In order to diminish the influence of widely distributed terms that might also be present in initial queries, a sample of the 2,000 most distributed terms in the target corpus was used as a common distractor space for all queries, as noted previously. Essentially, the term space for each query was expanded by the sampled 2,000 terms. Common distractor terms were used only in the initial search. In the final query, they were replaced by the distractor terms found specific to each individual query.

3.4.4 Automatic Feedback

The main issues for the automatic feedback loop include (1) selection of documents for positive feedback, (2) selection of documents for negative feedback (distractor space), (3) selection of terminology for enhancing the query, and (4) selection of distractor terminology for the query. The two experiments, CLARTN and CLARTF, differ in the way the documents for positive feedback were selected.

For CLARTN processing, the positive feedback was based on the top 50 document windows retrieved for initial queries. Document windows were processed using the thesaurus extraction; terms above the 60% level were added to the query. For CLARTF processing, the system used the filters (prepared for each query) to determine which among the top-100 document windows would be used for query augmentation. In fact, a maximum of 50 top document windows that satisfied the constraints was used in the thesaurus extraction process; again, terms above the 60% level were added to the query.

Distractor terminology for individual queries was identified using the lowest 1,500 scoring document windows retrieved by the initial query. Terms above the 70% thesaurus level were added to the query with an importance coefficient of 0. Their presence in the query term space reduced the score of documents that contained them.

3.4.5 Query/Document Matching

Cosine distance was used to measure similarity between queries and documents. To facilitate partial term matching, query terms were decomposed into attested subterms at the time of retrieval. The final score assigned to each document was the weighted average (d20-w1) between the similarity score of the best document window and the similarity score of the full document.

4 Analysis of the Retrieval Results

The official results of the TREC-4 ad-hoc query runs for CLARTN and CLARTF are given in Figure 7. Query-by-query performance for both systems (and compared to group performance) is given in Figures 8 and 9.

Comparison of the CLARIT–TREC-4 System precision/recall results with the median, minimum, and maximum system performance for individual queries calculated over all the TREC-4 ad-hoc experiments, given in Table 4, shows that CLARIT performed quite well, generally above the median.

4.1 Comparative Performance of CLARTN and CLARTF

The precision and recall statistics for individual queries in CLARTN and CLARTF experiments show that filtering windows for feedback has various effects on retrieval. For 41% of all queries, document filtering improves the average precision; for 35% filtering does not have any influence; and for remaining 24% it has a negative effect.

R-precision, on the other hand, remains the same for 53% of all queries. It is improved by document filtering for 27% of the queries and reduced for 20%. Comparison of document recall statistics shows that for 82% of all queries the recall increases or remains the same (41% of the queries in each category) when filtering is used.

The relative effect of document filtering on the system's average precision and recall proves to be quite modest. The precision/recall results presented in Figures 7, 8, and 9 show that this type of 'user assisted' feedback has a positive effect on precision and recall but, at the same time, does not achieve a statistically significant improvement. The same can be seen Table 5, which contains absolute and relative improvements, averaged over all queries. Graphs in Figures 10, 11, and 12 contrast the effect of filtering vs. not filtering on results for individual queries.

5 Future Experiments

Using the CLARIT IR system to explore TREC topics, we find that initial query formulation can have a range of effects on the outcomes of automated processes in the system, such as automatic feedback and document ranking. We notice that retrieved document sets generally converged towards a 'stable' document collection when automatic query augmentation is applied iteratively to the user's initial query. The convergence patterns, however, differ significantly across the queries and among different initial formulations of the same query.

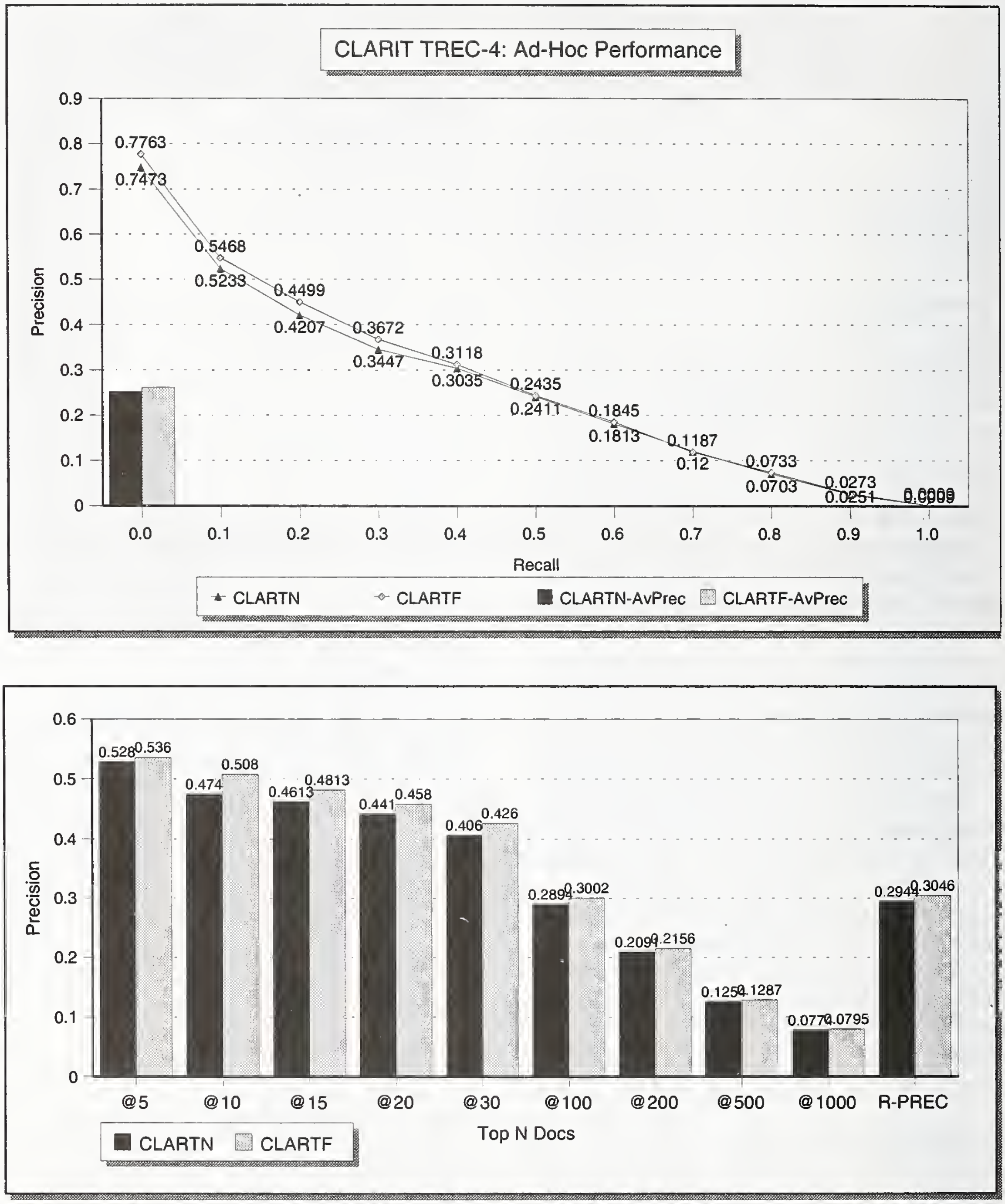


Figure 7: Official CLARIT–TREC-4 Ad-Hoc Query Results

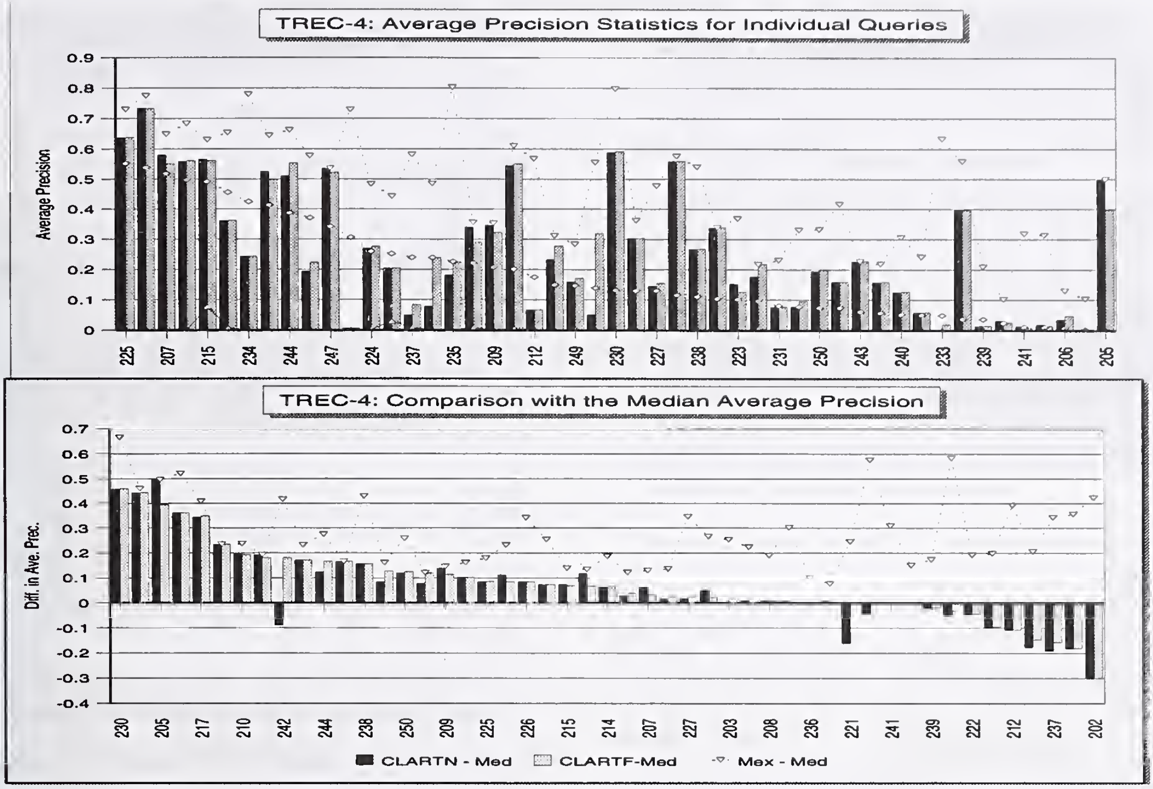


Figure 8: Query-by-Query Comparative Precision Results

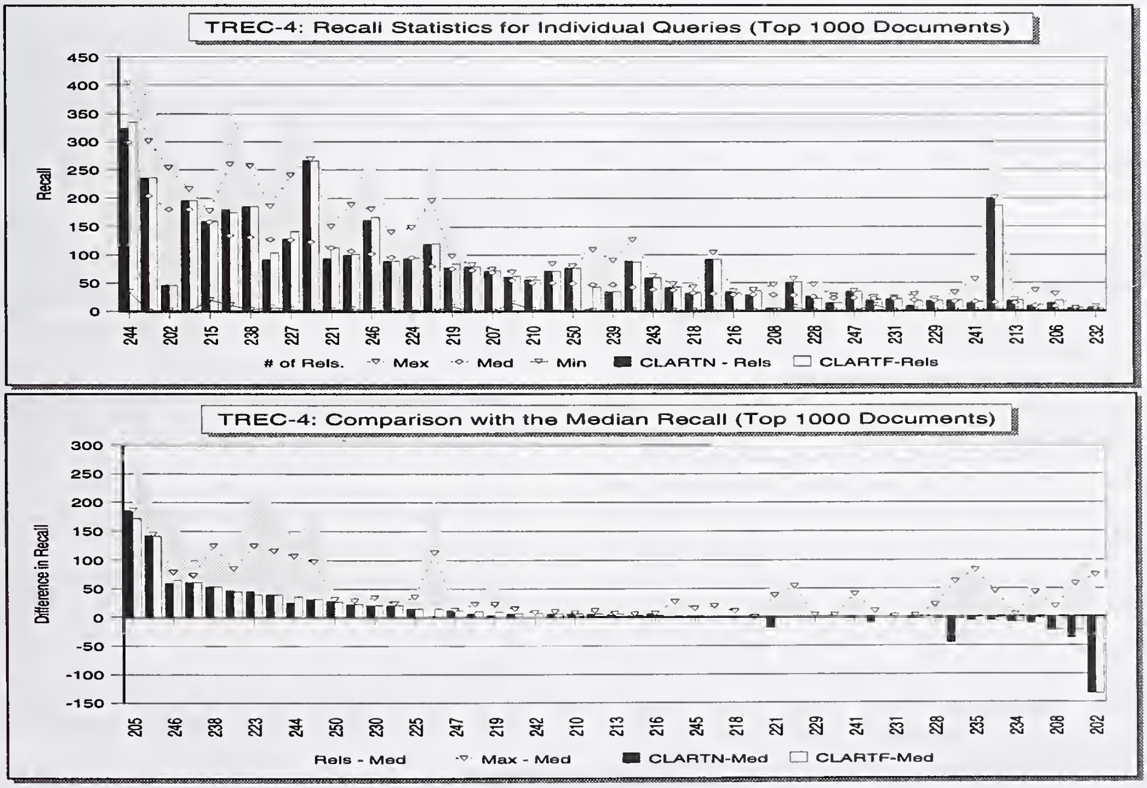


Figure 9: Query-by-Query Comparative Recall Results

	> Median		= Median		< Median	
	N	F	N	F	N	F
Precision	33	34	3	4	13	11
Rels in 1000	28	29	9	10	12	10

Table 4: Comparative Effects of CLARTN and CLARTF

Improv.per Q	**Avg.Prec.**	**R-Prec.**	**Recall**
Avg.Abs.Imp.	**0.0039**	**0.0049**	**1 Doc**
Avg.Rel.Imp.	**6.75%**	**3.73%**	**4.03%**

Table 5: Absolute & Relative Improvement with Filtering

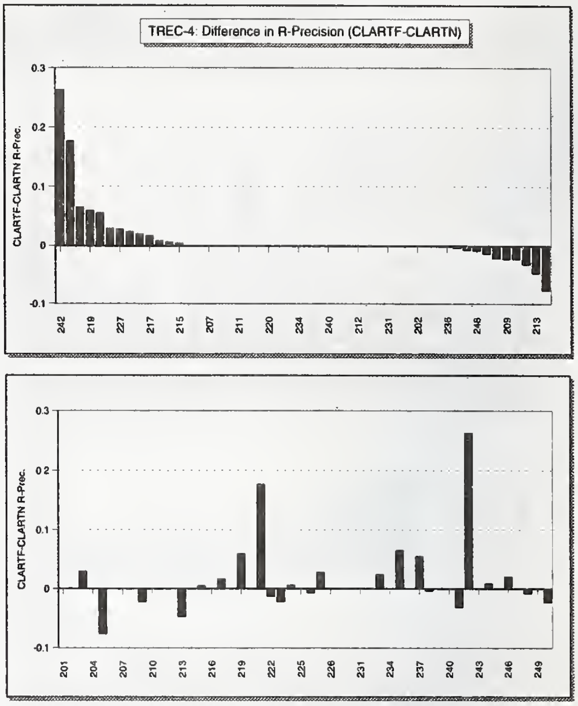


Figure 11: Comparative R-Precision: CLARTN/F

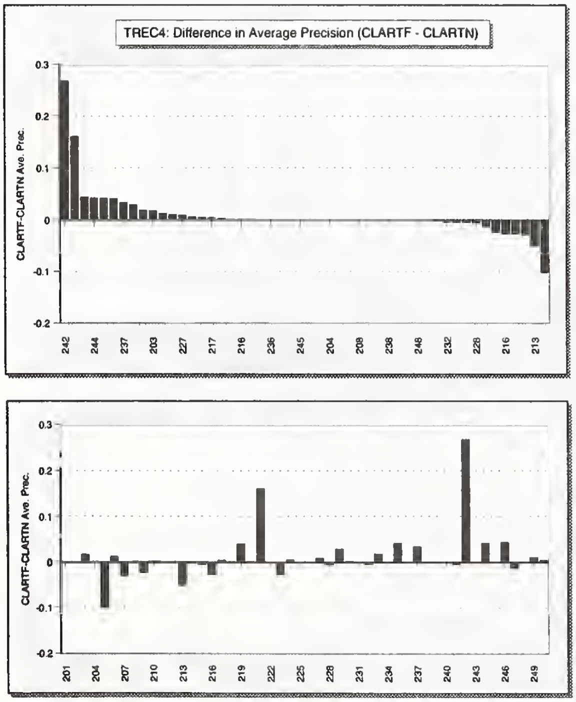


Figure 10: Comparative Average Precision: CLARTN/F

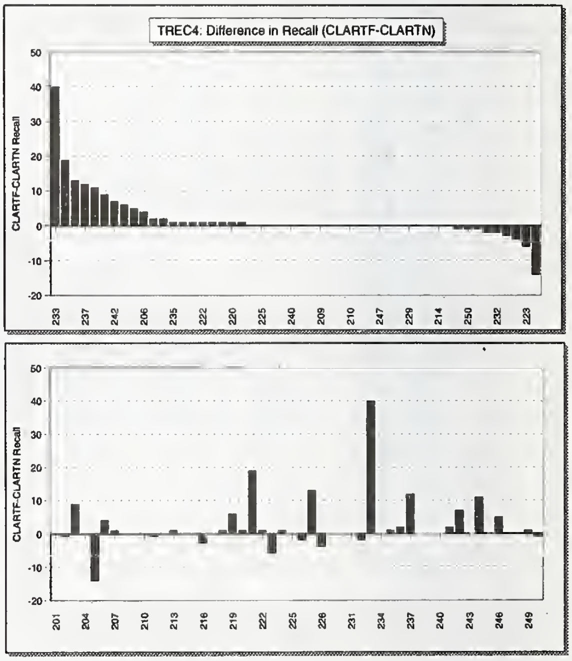


Figure 12: Comparative Recall: CLARTN/F

We would like to understand better the complex interaction between intrinsic or corpus-related characteristics of the query (e.g., level of specificity, representation of the topic in the target corpus, etc.), various properties of the user's formulation of a query (number of terms, decomposition into finer linguistic constituents, etc.), and different retrieval configurations of the CLARIT system. In a series of TREC-4 follow-on experiments, we have decided to explore first the effects of two factors:

1. the source of terminology used by the user for formulating the initial query; and
2. the level of control that the user has over the system's expansion of the query.

Based on the source of terms for the query, we will classify queries into three categories:

1. *Queries created automatically via NLP of the topic description*

   Terms in such queries will derive directly from CLARIT NLP of the topic description or information request. As a sub-category, we shall explore queries consisting of a subset of the NLP-generated terms (e.g., a few phrases) selected by the user to serve as 'seed' terms for the iterative enhancement process.

2. *Manually created queries supplemented with terms from a non-target corpus*

   Users will create the queries using the CLARIT IR system to find useful terms from information sources other than the target corpus.

3. *Manually created queries supplemented with terms from the target corpus*

   Users will create the queries using the CLARIT IR system over the target corpus.

Control of the feedback procedure will be realized at four different levels:

1. *Fully automatic query augmentation*

   The top N document windows retrieved by the initial query will be used by the system to extract a first-order CLARIT thesaurus. A specified portion of thesaurus terms will be added to the initial query automatically.

2. *Filtering of feedback documents based on the user's specified constraints*

   The user will specify a set of mandatory terms to be used to filter document windows retrieved by the initial query. Only the top N document windows that meet user's constraints will be used in thesaurus extraction. The specification of constraints will be made in advance, without any information about the target corpus.

3. *User selection of feedback documents*

   The user will be allowed to review documents that are retrieved by the initial query and select the ones to be used for automatic enhancement of the query.

4. *User selection of feedback documents and terminology to be added to the query*

   The user will be allowed to review documents retrieved by the initial query, select the documents for the thesaurus discovery process, and select terms to be added to the query.

As can be seen in Table 6, which summarizes the design of the set of experiments, there are twelve cases of interest, ranging from no user intervention at any point in the retrieval process (fully automatic processing, A1) to full user control at all points in the process (I4). Two of the cases have been addressed by the CLARTN and CLARTF runs.

Experiments designated by "I" will use queries created manually via interaction with the CLARIT IR system over the target database, assisted as appropriate by available relevance judgments. The results of these experiments will provide us with valuable information about the effectiveness of the distractor term space that is used in the CLARIT–TREC-4 system but not currently implemented in the CLARIT IR system.

Cearly, an important objective of experiments A3–M2–I3 and A4–M4–I4 is to test the system's ability to perform automatic feedback with the user's assistance in identifying truly relevant documents. Given the relevance judgments provided by NIST reviewers, we can approximate the outcome of two of the experiments, viz., M3 and M4, by simulating the interaction with a user who can reliably select truly relevant documents. In our first TREC-4 follow-on experiments, we have done just that.

To simulate M3, we used the same parameter settings and other procedures in the CLARIT–TREC-4 system, but automatically selected for feedback just those (at most fifty) top-scoring document windows that derived from known-relevant documents in the retrieved set of the top 100 documents returned by the initial query. We also used a distractor space consisting of the top 1,500 windows from known non-relevant documents, instead of a space deriving from the 'bottom' 1,500 windows. Note that such a choice for distractor documents represents the best contrast set for query refinement (and relevant/non-relevant document discrimination). Such a choice is realistic, since a set of

Query Types	Feedback Control			
	Fully Automatic	Document Filtering	Partial User Control (Docs)	Full User Control (Terms)
NLP Query	○ A1	○ A2	○ A3	○ A4
Terminology from Non-Target Databases	● CLARTN	● CLARTF	○ M3	○ M4
Terminology from the Target Database	○ I1	○ I2	○ I3	○ I4

Table 6: A Set of Future CLARIT Experiments

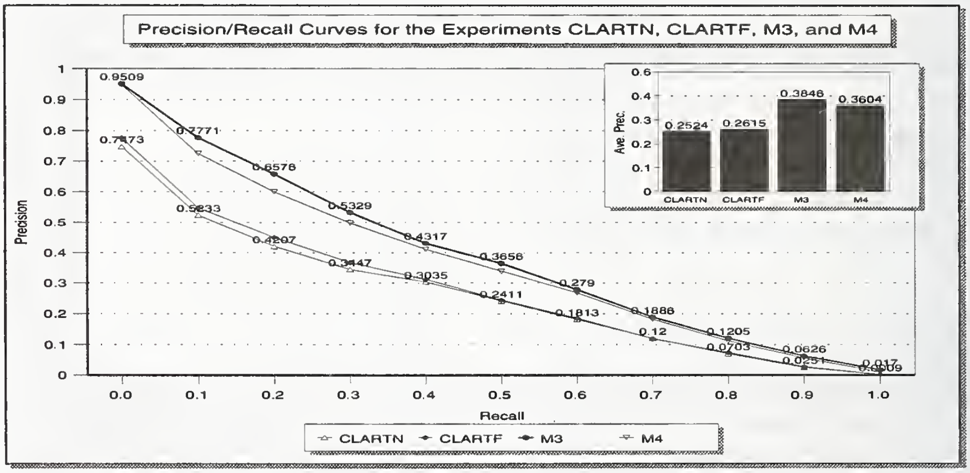


Figure 13: Precision/Retrieval Results for Varying Degrees of User Control

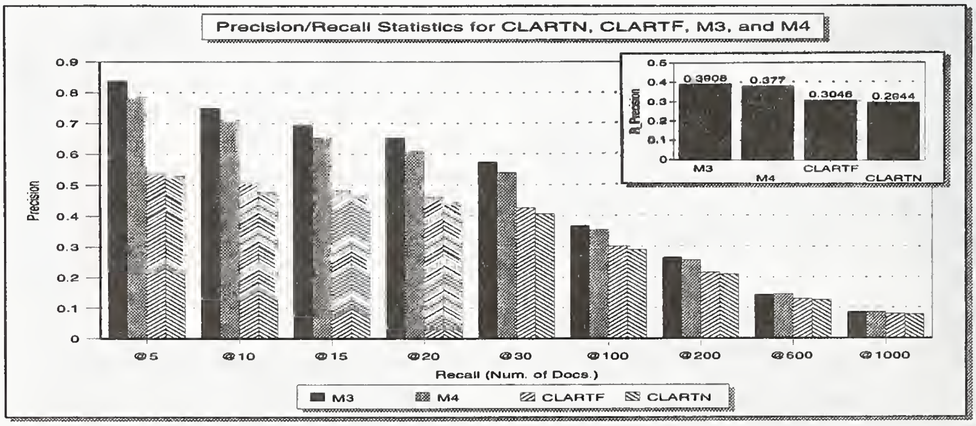


Figure 14: Graded Precision Results for Varying Degrees of User Control

non-relevant documents would be available to the system in any actual case where the user actively selects relevant documents. (The ones not chosen by the user could be treated as non-relevant.) The positive feedback terms and the distractor terms were added to the source-query terms; the resulting vector was used to retrieve a final ranked set of documents.

To simulate M4, we allowed users to review the feedback terms (positive and negative) that were generated by the first phase of M3 processing. In fact, we had two users review feedback terms for each query. The terms chosen by each were merged and used to supplement the source query.

Figures 13 and 14 show the results of the simulated M3 and M4 runs compared to the CLARTN and CLARTF ones. The four sets of results give a complete picture of the effects of varying degrees of user control. Both M3 and M4 achieve significant performance improvements over the CLARTN and CLARTF baselines, with the best performance by M3.

Results such as these suggest that CLARIT–TREC-4 system mechanisms such as thesaurus discovery to identify positive and negative terms and the use of query-independent term spaces are very powerful and can be used to support users in interactive retrieval. The techniques are designed, in particular, to require minimal intervention by the user and to be easy and natural to use—to involve little more than the normal steps a user engages in doing retrieval: formulating a query (in natural language) and making relevance judgments on a set of the top-most documents returned by the system in first-pass processing. Such results also demonstrate that we can achieve the goal of translating the lessons of TREC into practical value for users working interactively with the CLARIT system.

6 Acknowledgments

We would like to thank all the members of the CLARIT team for their assistance in conducting the TREC-4 experiments, in particular Cheng-Xiang Zhai, Xiang Tong, and Michael Mastroianni. We also would like to thank our colleagues at CLARITECH Corporation, who encouraged and supported our efforts in countless ways.

7 References

Buckley, C., Salton, G., Allan, J., & Singhal, A. Automatic query expansion using SMART: TREC-3. In D. Harman (Ed.), *Overview of the Third Text REtrieval Conference (TREC-3)* NIST Special Publication 500-225. Washington, DC: U.S. Government Printing Office, 1995, 69–80.

Callan, J.P., & Croft, W.B. An evaluation of query processing strategies using the TIPSTER collection. In R. Korfhage, E. Rasmussen, & P. Willet (Eds.), *Proceedings of the Sixteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, NY: Springer-Verlag, 1994, 292–300.

Efthimiadis, E.N., & Biron, P.V. UCLA-Okapi at TREC-2: Query expansion experiments. In D. Harman (Ed.), *The Second Text REtrieval Conference (TREC-2)*. NIST Special Publication 500-215. Washington, DC: U.S. Government Printing Office, 1994, 279–290.

Evans, D.A., Lefferts, R.G., Grefenstette, G., Handerson, S.K., Hersh, W.R., and Archbold, A.A. CLARIT TREC design, experiments, and results. In D. Harman (Ed.), *The First Text REtrieval Conference (TREC-1)*. NIST Special Publication 500-207. Washington, DC: U.S. Government Printing Office, 1993, 251–286; 494–501.

Evans, D.A., & Lefferts, R.G. Design and evaluation of the CLARIT–TREC-2 system. In D. Harman (Ed.), *The Second Text REtrieval Conference (TREC-2)*. NIST Special Publication 500-215. Washington, DC: U.S. Government Printing Office, 1994, 137–150.

Evans, D.A., & Lefferts, R.G. CLARIT–TREC experiments. *Information Processing and Management*, 1995, Vol. 31, No. 3, 385–395.

Milić-Frayling, N., Zhai, C.-X., Tong, X., Mastroianni, M.P., Evans, D.A., Lefferts, R.G. (1996). CLARIT TREC-4 interactive experiments. In D. Harman (Ed.), *The Fourth Text REtrieval Conference (TREC-4)* NIST Special Publication. Washington, DC: U.S. Government Printing Office.

CLARIT TREC-4 Interactive Experiments

Natasa Milic-Frayling[1], Cheng-Xiang Zhai[2], Xiang Tong[2]
Michael P. Mastroianni[1], David A. Evans[1,2], Robert G. Lefferts[1]

[1]CLARITECH Corporation
319 South Craig St., Suite 200
Pittsburgh, Pennsylvania 15213-3726

[2]Laboratory for Computational Linguistics
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213-3890

1. Introduction

The main stream TREC tasks [1]-[3], ad-hoc and routing experiments, are aimed at qualifying the performance of retrieval systems operating in a fully automatic mode, with or without user's assistance in formulating initial queries. Accordingly, the design and evaluation of these experiments are primarily concerned with the characteristics of the underlying search engine. The TREC Interactive task [3], on the other hand, models a complex retrieval situation in which the user is allowed to make decisions and influence the retrieval procedure through available modes of interaction with the system. As such, this task provides an opportunity for identifying and exploring factors that influence the quality of information retrieval in a fully interactive environment.

The CLARIT Interactive system used in the TREC-4 experiments provides the searcher with a high level of control over the main retrieval processes: formulation of the query, modification and fine tuning of the query, and selection of relevant documents. Consequently, the retrieval process is influenced to a great extent by the searching style and relevance judgments of a particular user. This significantly increases the complexity of the retrieval evaluation.

In fully automatic TREC experiments, evaluation is relatively straightforward. A topic that represents an expert's information need is automatically transformed into a query compatible with the underlying search engine and executed over TREC databases. The official TREC precision/recall evaluation [1]-[3], based on expert judgments, provides useful information about a system's ability to represent experts' information needs and retrieve relevant documents automatically. In the manual category of TREC experiments the meaning of such an evaluation is less transparent. These experiments involve a subjective interpretation of the expert's topic by a searcher whose goal is to formulate a query that accurately reflects the expert's information need, complies with the system requirements, and optimizes the retrieval results. In such a retrieval situation, an evaluation metric based on expert judgments incorporates the evaluation of both the retrieval system and the particular user's ability to create appropriate queries.

The interactive task presents a serious challenge in terms of evaluation and experimental design. Not only is the effectiveness of the initial query formulation compounded with the performance of the automated processes in the system, but so are the effects of all intermediate steps that require or allow for user judgment. Furthermore, while reliance on an authoritative expert judgment provides a useful evaluation measure for fully automatic retrieval, the inherent dichotomy between the 'real user', the person performing the search, and the 'expert user', the person judging the results, makes it impossible to evaluate certain types of interactions between the user and the system (e.g., relevance feedback).

In our analysis of the CLARIT TREC-4 Interactive experiments we adopt the evaluation metric based on expert judgments (micro/macro average precision and recall) as a useful reference point for characterizing the user's understanding of the topic and relevance criteria. However, we introduce a number of additional precision and recall measures to help us delineate the influence of various factors involved in the interactive retrieval process. In particular, we analyze the effects of three important search components: (1) the interpretation of the topic and the implicit relevance criteria of the user, (2) the time limit on the search session, and (3) the 'reliability' of documents used for CLARIT automatic feedback. We base our analysis on a sequence of experiments and measurements that gradually relax the restrictions and limitations of the single experiment design; they essentially eliminate the time constraint on the search session and reduce the discrepancy between the user's and the expert's judgment.

In the following sections we describe the CLARIT Interactive System (Section 2), the design and the official results of the CLARIT Interactive task (Section 3), an analysis of the official primary interactive task (Section 4) and subsequent secondary interactive runs (Section 5), and we present our concluding remarks (Section 6). The Appendix contains the TREC standardized system description of the Interactive CLARIT system.

2. CLARIT Interactive System

The CLARIT TREC-4 Interactive experiments were performed using Version 3.0B4 of the CLARIT[1] software on the DEC Alpha computer platform, which incorporates the main IR techniques of the CLARIT system, in particular, the following:

- **Natural language processing (NLP) of queries and documents**

  The two main components of the CLARIT NLP [4] are an inflectional morphological analyzer used for word recognition and normalization and a deterministic, ruled based parser for phrase identification. Although the NLP module is capable of providing complex linguistic constituents, we generally use simplex noun phrases for document indexing and query processing. The CLARIT NLP module is engineered for text processing at the rate of 200 MB per hour (3.5 MB/min). This processing speed is sufficient to support a number of advanced

[1] In the TREC-4 Interactive experiment we used the standard CLARIT IR system that has been commercialized by CLARITECH Corporation. This system differs significantly from the CLARIT-TREC System, the experimental suite of tools that has been used in TREC ad-hoc and routing tasks.

CLARIT features, such as CLARIT Thesaurus Discovery, Document Summarization, and highlighting of features in documents, to be implemented as run-time processes.

- **Document retrieval and ranking based on a vector space retrieval model**

CLARIT Indexing. Document retrieval is facilitated by inverted indices prepared using the CLARIT Indexing facility. The CLARIT Indexing module processes documents at the rate of 80-100 MB of text per hour. This rate includes all the processing necessary to build a CLARIT database (e.g., the CLARIT NLP of the text). The size of the resulting index ranges from 40-85% of the original data size. The relative size of the indexing overhead is smaller for larger databases. Furthermore, the overhead increases with the number of fields in the documents for which the indexing is done. For illustration, a database of 56 MB indexed over two common fields, document title and body of the text, may require 42 MB (75% of the source).

Inverted indices can be created for full documents or for parts of the documents; we commonly index on sub-documents of an approximately fixed size. This feature provides for document normalization, which enables us to use simple feature matching as a similarity measure between queries and documents (the inner product of the query and the document term vectors). Final document ranking can be done in various ways; for this purpose, CLARIT 3.0B4 uses the score of the best matching sub-document.

CLARIT Querying. Query preparation for retrieval includes (1) CLARIT NL processing of the query text to extract the query terms (phrases), (2) decomposition of the query terms into linguistic substructures (single-word terms, attested sub-terms), and (3) scoring of the query terms against the inverted indexes.

In general, the speed of querying in the CLARIT system depends on the query size and the distribution of query terms in the target corpus. However, the design of the CLARIT Search Engine provides for optimization of query time by using a caching mechanism that restricts the number of documents considered in a particular search. The 'working set' used in the search consists of documents (or sub-documents) that contain the most discriminating terms in the query. Practically speaking, the documents for the working set are selected based on the distribution count of query terms, starting with those that contain a query term with the lowest distribution count. The user can adjust the size of the working set to achieve a desired speed of retrieval. The default size of the working set is 2,500 sub-documents, a setting that has been shown to be appropriate for common database searching. If the working set size is not specified the system searches over all documents in the database.

For a typical database of approximately 250MB and a working set of 2,500 sub-documents, a query of 7 concepts (about two sentences) requires 0.25 seconds, a query of

100 concepts (about a paragraph) requires 0.5 seconds, and one with 800 concepts (approximately the size of a news article) executes in 1.6 seconds.

The CLARIT Search engine supports simultaneous querying over multiple databases by treating the databases as a single corpus of documents. Document retrieval and ranking are based on the unified database statistics.

- **Automatic discovery of characterizing terminology for a set of documents**

The CLARIT Thesaurus Discovery technique generates the first-level thesaurus, essentially a list of statistically and linguistically prominent terminology from a selected set of documents. Selection and ranking of thesaurus terms is generally based on an analysis of linguistic structure and associated term distribution patterns in the document set, but it can also incorporate statistical information about general frequency of the term in the English language or in the specific topical domain. Thesaurus Extraction is used as a basis for automatic or user assisted query augmentation [4] and for concept expansion over a selected CLARIT database.

The user interface for Version 3.0B4 of the CLARIT software is in fact a gui intended for demonstrating system search capabilities (see Appendix). It therefore includes all the main retrieval features of the CLARIT system. However, it was not designed to record a detailed search history automatically, such as a user's keystrokes, which would be useful for the analysis of user interaction with the system. Some of the difficulties in collecting data caused by this deficiency were overcome by video recording of the search sessions, and by modifying standard system features. For example, the 'Search' button, typically used to submit a query for retrieval, was enabled to store the query formulation and details about retrieved documents automatically. Although we could not capture all the interesting details of the interaction between the searcher and the system, the recorded data provided valuable insights about the relevant properties of a fully interactive search environment.

3. TREC-4 Interactive Tasks

3.1 Description of the tasks

The TREC-4 Interactive experiments consist of two tasks. In the primary, the searcher interacts with the system with the goal of retrieving as many relevant documents as possible for a given topic within a 30 minute time period. Documents retained by the searcher are evaluated for precision and recall by experts at NIST. In the secondary interactive task, the final queries created during the primary task experiments are used to obtain the top 1000 documents retrieved by the system. The retrieval results are evaluated using the standard TREC precision and recall statistics.

For the official TREC-4 evaluation we submitted one full set of search results for the primary interactive task, the CLARITI run, which was obtained by a single searcher. The second set of interactive experiments, performed by two other searchers, was incomplete and therefore not submitted for official evaluation. However, we found that all three searchers exhibited interesting characteristics, which we discuss in Section 4. Furthermore, because of time constraints, we were unable to submit for official evaluation the results of the secondary task. In Section 5 we present our own evaluation of subsequently performed secondary task experiments.

3.2 Experiment Set-up

The two sets of CLARIT TREC-4 Interactive experiments involved three searchers:

- Searcher A, a person familiar with the CLARIT IR System and CLARIT gui, but with no training and experience in professional searching.
- Searcher B, a professional searcher with limited experience in using the CLARIT IR System
- Searcher C, a person new to electronic data searching and using the CLARIT IR System.

Prior to the experiments searchers B and C were given a 35-45 minute guided tour of the CLARIT user interface. They performed sample searches using most of the retrieval features available through CLARIT gui. In particular, they learned how to

- formulate a query
- view the query vector
- adjust the importance coefficients of query terms
- submit the query for search
- view the list of retrieved documents
- view the text of retrieved documents
- keep relevant documents and discard non-relevant documents
- select documents for query augmentation and view the extracted CLARIT thesaurus
- add terms from the CLARIT thesaurus to the query
- use external databases to identify additional terminology for query expansion
- save the final query and the final result set
- end the search session.

For initiating a search on a given topic the searcher has the option of downloading the full description of the topic or typing in his or her own query. Modification of the query is possible in a number of ways:

- by manually adding terms to the query
- by adding terms from a CLARIT Thesaurus discovered from selected documents in the target database
- by adding CLARIT Thesaurus terms extracted from external databases.

The searcher can also manually adjust importance coefficients of individual terms in the query.

At each search iteration the top 150 retrieved documents were presented to the searcher for inspection. The CLARIT gui uses separate windows for viewing a list of retrieved documents (with documents titles and ranking information) and for reading the document text. Viewing of the document text is made more efficient by highlighting all the full noun phrases in the text that contain any single-word sub-terms of query terms (phrases). In addition, the user can have the document window automatically scrolled to the best scoring paragraph (sub-document) in the text.

Furthermore, the gui for the experimental CLARIT system was modified to keep track of documents automatically for which the full text was viewed. Relevance information about documents was recorded via the Keep and Discard functions: the searchers marked as Kept all the documents that they found relevant and as Discarded all the ones they found non-relevant for the query.

Preparation of data for the interactive experiments involved 'claritizing' 1.77 GB of TREC-4 data into a CLARIT database with an inverted index of 1.02 GB (58% of the size of the source). Since the data was processed using simple heuristics for identifying document titles that did not take into account various document formats in the TREC databases, some of the documents were designated as "*<no title>*" documents. We therefore encouraged the searchers to view document text in order to establish the relevance of the retrieved documents.

4. CLARIT Interactive Experiments

4.1 TREC-4 official results

The CLARIT TREC-4 interactive run, CLARITI, was performed by searcher A. Table 1 contains the official TREC-4 results for the CLARITI experiment. The precision statistic presented in the table is simply the percentage of documents selected by the searcher that were judged relevant by NIST experts. The recall statistic is calculated as the percentage of all relevant documents for a topic that were detected by the user during the search session.

The scatter-plots for precision and recall presented in Figure 1 clearly show that searcher A was more concerned with quality than with the quantity of the retrieved documents: for 18 out of 25 topics (72%) the searcher kept fewer than 20 documents. On the other hand, for 17 out of 25 topics (68%) the retrieval precision is above 0.6 which results in a micro average precision of 0.67 and a macro average precision of 0.70. At the same time the recall statistics never exceed 0.5.

We are inclined to interpret the micro/macro precision figures as the level of agreement between the expert's and the searcher's notion of relevance rather than as statistics that accurately describe the global precision of the complex User-CLARIT system. Indeed, such statistics do not capture important characteristics of the search process such as the order in which documents are presented to the user, the number of search iterations, the number of relevant documents retrieved in each search iteration, and other features of the User-CLARIT interaction. Therefore they cannot be used to answer important questions about the system performance such as how the recall and precision statistics are influenced by the searcher's selectivity in keeping relevant documents on one hand and the system's ability to retrieve relevant documents on the other.

SEARCHER A - CLARITI Experiment					
Query ID	Tot_Rel	Retrieved	Rel_ret	Precision	Recall
202	283	24	19	0.79	0.07
203	33	3	1	0.33	0.03
204	397	1	1	1	0
205	310	9	9	1	0.01
214	47	5	2	0.4	0.04
207	74	33	26	0.76	0.34
208	54	18	3	0.87	0.76
208	87	11	3	0.36	0.05
210	57	23	20	0.87	0.35
211	323	3	3	1	0.01
212	153	26	28	0.88	0.15
214	21	3	2	0.67	0.4
214	5	1	1	1	0.2
220	153	61	53	0.67	0.29
214	36	41	17	0.38	0.67
220	21	5	9	0.8	0.67
223	363	9	6	0.67	0.02
227	347	30	28	0.88	0.88
232	9	1	0	0	0
236	43	3	0	0	0
238	270	14	9	0.64	0.03
239	123	11	5	0.45	0.04
242	38	1	1	1	0.03
243	69	8	5	0.63	0.07
250	86	3	3	1	0.03
Micro Av.	137	14	10	0.67	0.11
Macro Av.				0.7	0.07

Table 1. Precision and recall statistics for the CLARITI Experiment

For example, it would be important to identify the source of the significant discrepancy between the average number of retrieved documents, 14 documents per topic, and the average number of relevant documents, 137 per topic, observed in the CLARITI experiment (Table 1).

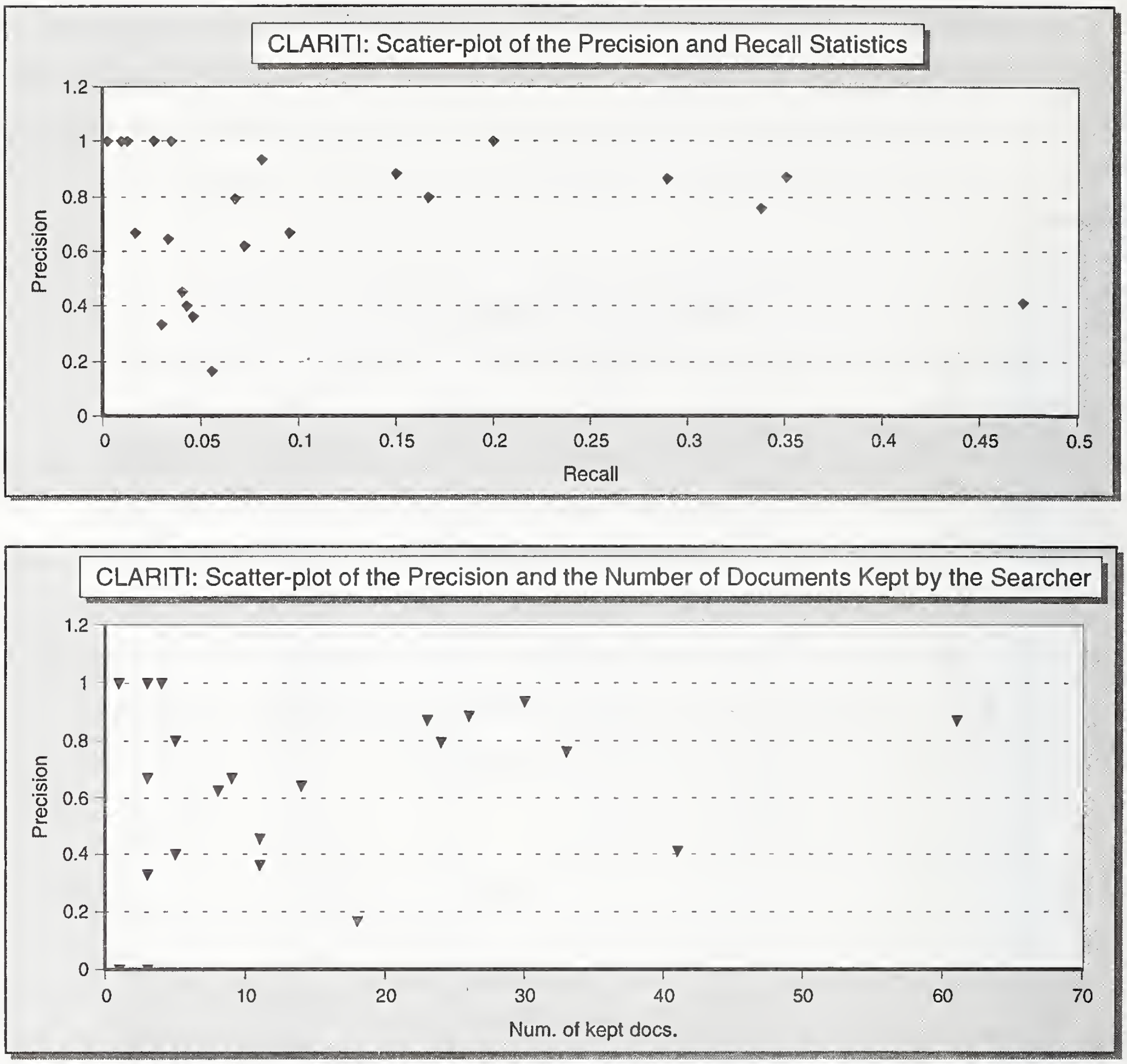


Figure 1: The scatter-plots of the precision against the recall and the number of kept documents, respectively

These and related issues motivate the detailed analysis of CLARITI results that we present in the following section.

4.2 Analysis of CLARITI retrieval results

During the CLARIT interactive search session the system stored intermediate search results, which enabled us to analyze three important factors affecting retrieval outcomes:

- the user's understanding of the task, reflected in the document selection process
- the 30 minute time constraint on the search sessions
- CLARIT System retrieval performance, including automatic query modification based on user feedback (in particular, contrasting expert and non-expert feedback).

4.2.1 Influence of the user's relevance judgments

In the CLARIT system, the searcher's relevance judgments play two significant roles: first, as criteria for keeping relevant documents that will be reviewed by NIST experts, and second, as a mechanism for selecting documents that are fed back into the search process for automatic augmentation of the query. In this section we analyze the first aspect of the user's judgment; the second will be discussed in section 4.2.3 as a part of the analysis of relevance feedback.

As a composite precision measure of the User-CLARIT system performance we introduce P_K (kept documents precision), the non-interpolated average precision calculated over documents viewed by the searcher. More precisely, to obtain P_K for a topic we compute for each viewed document the ratio $\frac{\text{\# of kept relevant documents}}{\text{\# of viewed documents}}$ and average this statistic over all viewed documents and all search iterations for the topic:

$$P_K = Avg_{search}(Avg_{viewed\ docs} \frac{\text{\# of kept relevant documents}}{\text{\# of viewed documents}})$$

Here the relevance of the kept documents is established based on the relevance judgment of experts at NIST.

The precision measure P_K takes into account, to a certain degree, the order in which the documents were retrieved by the system, although this statistic is calculated only over the documents viewed by the user. Indeed, we observed that the searchers regularly reviewed documents starting from the top of the ranked list. This was true even in subsequent search iterations which contain discarded and kept documents; the searcher mostly reviewed newly retrieved documents as they appeared in the list. By calculating the precision for all documents viewed in all search iterations (thus re-counting the kept and discarded documents based on their ranking in individual search iterations) we capture both the CLARIT retrieval characteristics and the searcher's document selection pattern.

We use precision P_K to determine the degree to which a searcher's document selection criteria affect the precision and recall of the retrieval. We compare P_K with the precision that would be achieved by an expert user who is reviewing and judging the same set of documents. We calculate P_V (viewed documents precision) by computing the ratio $\frac{\text{\# of relevant documents}}{\text{\# of viewed documents}}$ for each document viewed by the user and determining the average over all viewed documents and all search iterations for a given topic:

$$P_V = Avg_{search}(Avg_{viewed\ docs} \frac{\text{\# of relevant documents}}{\text{\# of viewed documents}}) .$$

After eliminating the main outliers for the relative difference statistic (Table 2), we find that the discrepancy between the user's and the expert's document selection criteria leads, on average, a 180% relative difference in precision (Table 3).

Topic Num.	**P_K**	**P_V**	**Diff**	**Rel. Diff.**
203	0.0053	0.3919	0.3866	**7,300%**
204	0.0001	0.079	0.0789	**78,900%**
232	0	0.093	0.09	**undefined**
236	0	0.0053	0.01	**undefined**

Table 2. Outliers excluded from the relative difference statistics

	P_K		**P_V**		**Diff**		**Rel. Diff.**
	all topics	21 topics	all topics	21 topics	all topics	21 topics	21 topics
Mean	**0.12**	**0.14**	**0.24**	**0.25**	**0.12**	**0.11**	**180%**
Median	**0.07**	**0.09**	**0.17**	**0.22**	**0.09**	**0.09**	**126%**
Std Dev	0.12	0.12	0.16	0.16	0.1	0.09	224%
Max	0.33	0.33	0.57	0.57	0.39	0.37	943%
Min	0	0	0.01	0.01	0.004	0	2%

Table 3. Summary of precision statistics

Similar effects can be observed by comparing two recall measures: R_K, the percentage of relevant documents viewed and kept by the user, and R_V, the percentage of relevant documents viewed by the user:

$$R_K = \frac{\text{\# of (unique) kept relevant documents}}{\text{\# of relevant docs. for the topic}} \quad \text{and} \quad R_V = \frac{\text{\# of (unique) viewed relevant documents}}{\text{\# of relevant docs. for the topic}}.$$

After removing the main outliers for the Rel.Diff. statistic (Table 4), the recall statistics can be summarized as presented in Table 5.

Topic Num.	**R_K**	**R_V**	**Diff**	**Rel. Diff.**
203	0.0303	0.4848	0.4545	**1,500%**
203	0.0025	0.0277	0.0252	**1,000%**
211	0.0093	0.1827	0.1734	**1,865%**
232	0	0.56	0.56	**undefined**
236	0	0.02	0.02	**undefined**

Table 4. Outliers excluded from the statistical analysis

	R_K		**R_V**		**Diff**		**Rel. Diff.**
	all topics	20 topics	all topics	20 topics	all topics	20 topics	20 topics
Mean	**0.11**	**0.13**	**0.32**	**0.34**	**0.21**	**0.21**	**185%**
Median	**0.06**	**0.07**	**0.19**	**0.19**	**0.12**	**0.12**	**187%**
Std Dev	0.13	0.13	0.29	0.3	0.21	0.2	111%
Max	0.47	0.47	1	1	0.8	0.8	400%
Min	0	0.01	0.02	0.02	0.01	0.01	50%

Table 5. Summary of recall statistics

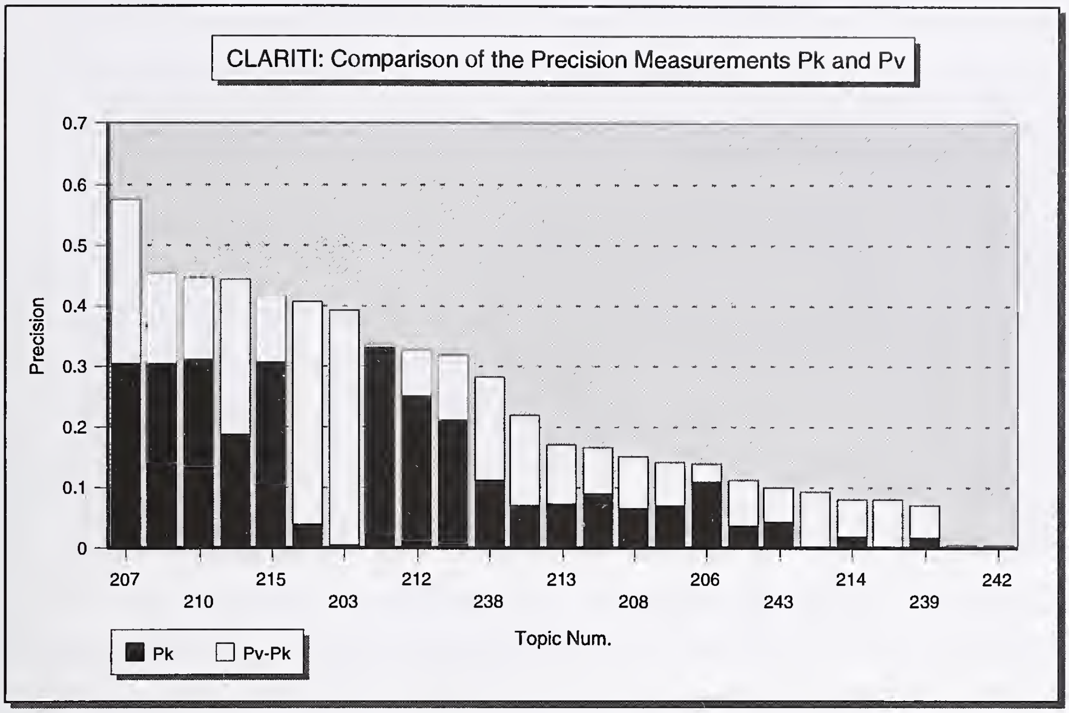


Figure 2. Effect of the user's relevance judgment on precision

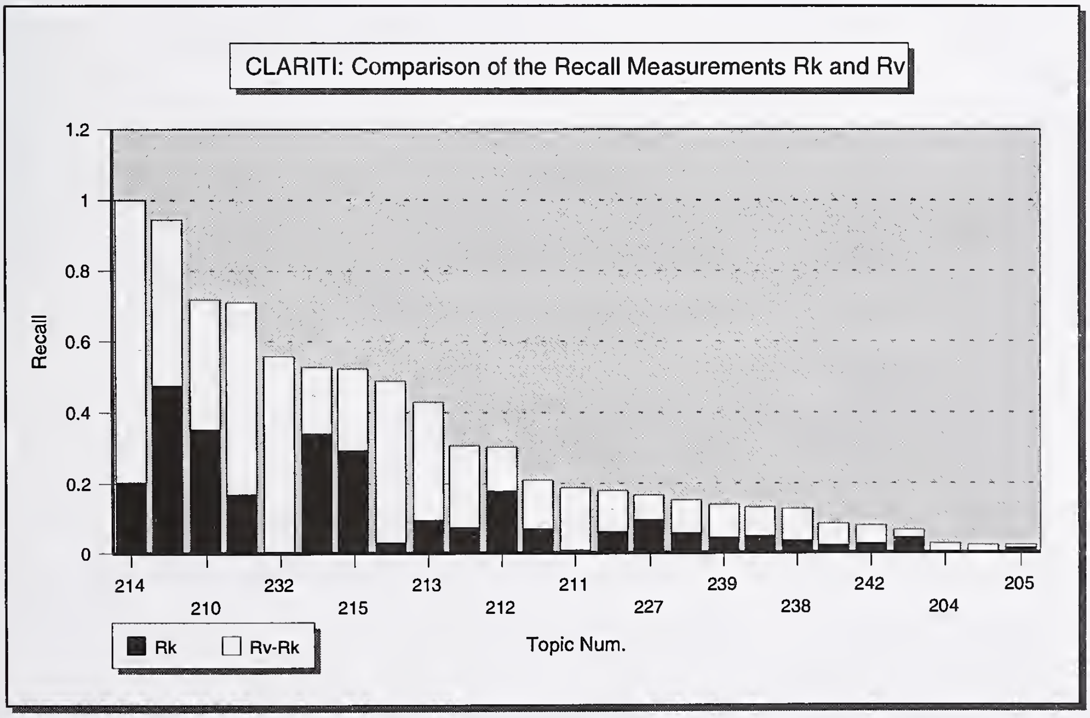


Figure 3. Effect of the user's relevance judgment on recall

The comparison of the recall statistics therefore shows that the searcher's selection criteria differs significantly from the expert's relevance judgments. If we use the expert's document judgments as the relevance criteria then the set of viewed documents contains on average 21% more relevant documents than identified by the searcher. This results in the relative increase in recall of 185% on average.

4.2.2 Influence of the time constraint

The time restriction on the search session increases the impact that the user's searching style has on the retrieval results. This is particularly true for systems such as CLARIT whose search engine is optimized for fast retrieval so that the time used by the system to execute a search is negligible compared to the time available to the user for manipulating a query text, reviewing and selecting presented documents and exploring different search strategies. (see Section 2). For example, a video recording of the search session for Topic 236 reveals that searcher A spent 10.7 minutes of the 13 minute search session (82% of the total time) viewing titles and document texts. The remaining 2.3 minutes were used by the searcher for modifying and submitting new queries, and by the system for loading the topic description, parsing queries, saving queries and search results, and presenting and highlighting the text of the viewed document. The total time used for execution of queries was on the order of seconds.

Unfortunately, the present design of our interactive experiments does not enable us to perform a comparative analysis of the retrieval results for searcher A with and without the 30 minute time restriction. However, based on the available relevance judgment of documents retrieved and viewed during the CLARITI search session we can estimate the impact that the time restriction would have for an expert user[2].

For this purpose we introduce a precision measure P_R (retrieved documents precision), based on all the documents presented to the user during the search session. In this manner we simulate the retrieval situation in which the expert user reviews and evaluates for relevance all 150 documents retrieved during each search iteration. More specifically, for each retrieved document we calculate the ratio $\frac{\text{\# of relevant documents}}{\text{\# of retrieved documents}}$ and average over the 150 documents presented in the search iteration. To calculated P_R for a topic we average these statistics over all search iterations:

$$P_R = Avg_{search}(Avg_{150}\frac{\text{\# of relevant documents}}{\text{\# of retrieved documents}}).$$

The graphs in Figure 5 illustrate the difference between P_R and the precision P_V calculated only over documents viewed during the 30 minute search sessions. The summary statistics in Table 6 show that an unconstrained search by an expert would lead to a 6% absolute increase and a 21% relative increase in precision on average.

[2] Here we tacitly assume that the expert has a similar searching style to that of searcher A, thus viewing the retrieved documents at the similar rate. Furthermore, we take a simplifying view that under no time restriction the expert user would evaluate for relevance all the documents returned by the system.

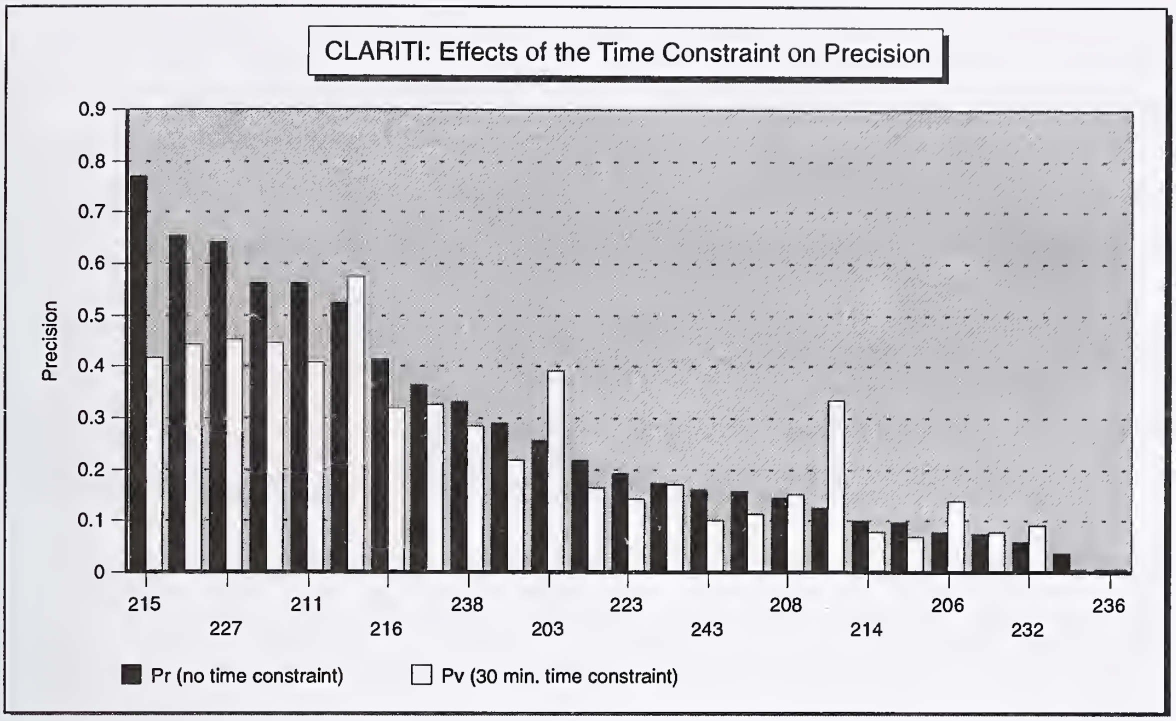


Figure 4. Comparison of the retrieval precision for the search with and without time constraint

	P_V		P_R		**Diff**		**Rel. Diff.**
	all topics	24 topics	all topics	24 topics	all topics	24 topics	24 topics
Mean	**0.24**	**0.24**	**0.28**	**0.28**	**0.04**	**0.06**	**21%**
Median	**0.17**	**0.19**	**0.19**	**0.19**	**0.04**	**0.06**	**30%**
Std Dev	0.16	0.22	0.22	0.16	0.11	0.11	135%
Max	0.57	0.77	0.77	0.57	0.36	0.35	669%
Min	0.01	0.01	0.01	0.01	-0.21	-0.21	-62%

Table 6. Summary of the precision statistics

Similarly, we can compare the recall measure R_V with R_R, a recall measure that represents the percentage of all relevant documents that have been presented to the user during the search session:

$$R_R = \frac{\text{\# of (unique) relevant documents retrieved}}{\text{\# of relevant docs. for the topic}}.$$

Again, the summary of recall statistics (Table 7) shows that the elimination of the time constraint results on average in a 7% absolute increase and an approximately 36% relative increase in recall for an expert user.

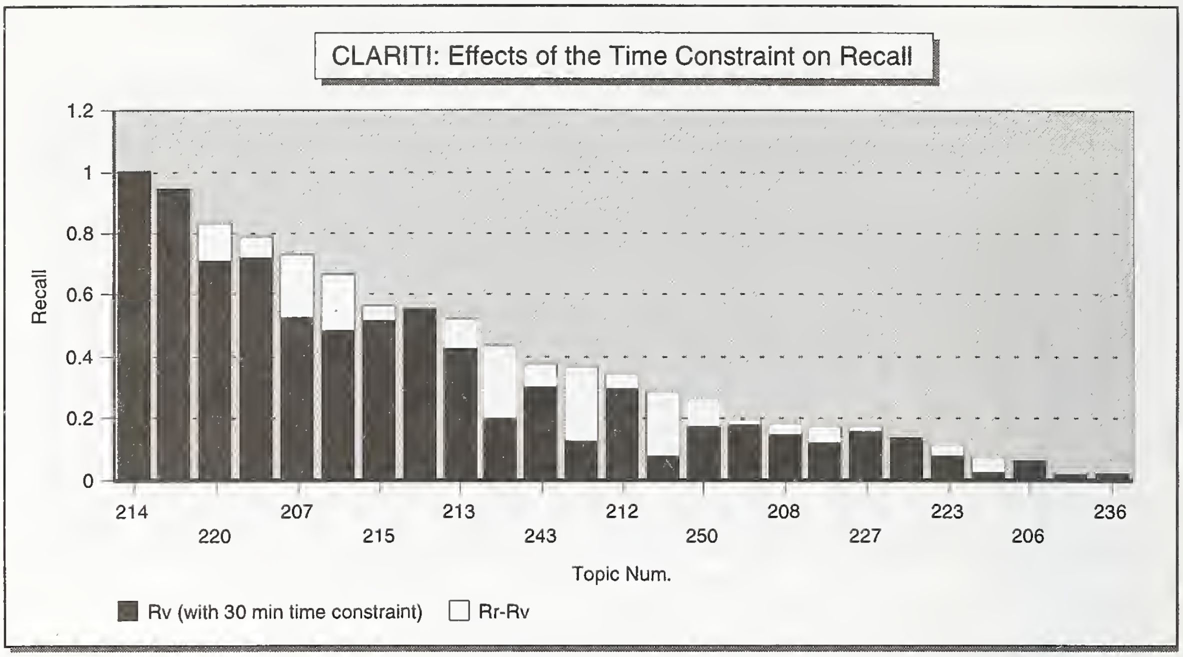


Figure 5. Comparison of the retrieval recall with and without a time constraint on the search session

	R_V		R_R		Diff		Rel. Diff.
	all topics	24 topics	all topics	24 topics	all topics	24 topics	24 topics
Mean	**0.32**	**0.33**	0.4	**0.4**	**0.21**	**0.07**	36%
Median	**0.18**	**0.19**	0.35	**0.36**	**0.12**	**0.04**	20%
Std Dev	0.29	0.3	0.3	0.3	0.21	0.08	51%
Max	1	1	1	1	0.8	0.24	191%
Min	0.02	0.02	0.02	0.02	0.01	0	0%

Table 7. Summary of the recall statistics

4.2.3 Influence of relevance feedback

One of the important features of the CLARIT Interactive system is its ability to discover related terminology from a given set of documents automatically and use that terminology to enhance the query. However, the effectiveness of this feature can be evaluated only through a full simulation of a search session in which the same relevance judgments are used for both the selection of documents for feedback and the evaluation of retrieved documents.

For this purpose we performed an additional set of interactive experiments, CLARIT-EFB, which includes two search iterations for each topic, both involving query augmentation based on retrieved relevant documents. In each search iteration we selected from the top 150 retrieved documents all those that were judged relevant by the NIST experts. From these documents we extracted CLARIT Thesauri and used the top scoring terminology to supplement the query. In principle, the searcher can decide to add to the query only the terminology that he or she perceives as useful for the search. In our simulation we restricted the expert's feedback to query augmentation involving all the terms above a certain thesaurus level. The selection of the thesaurus level was roughly based on the number of documents used for feedback and the number of terms in the thesaurus. An automated version of this selection mechanism could be easily implemented.

The retrieval results of CLARIT-EFB were evaluated using the precision measure P_R and the recall measure R_R. Comparing these results with the same statistics obtained from CLARITI, we can see that consistently applied automatic feedback with reliable relevance information has beneficial effects on both precision and recall.

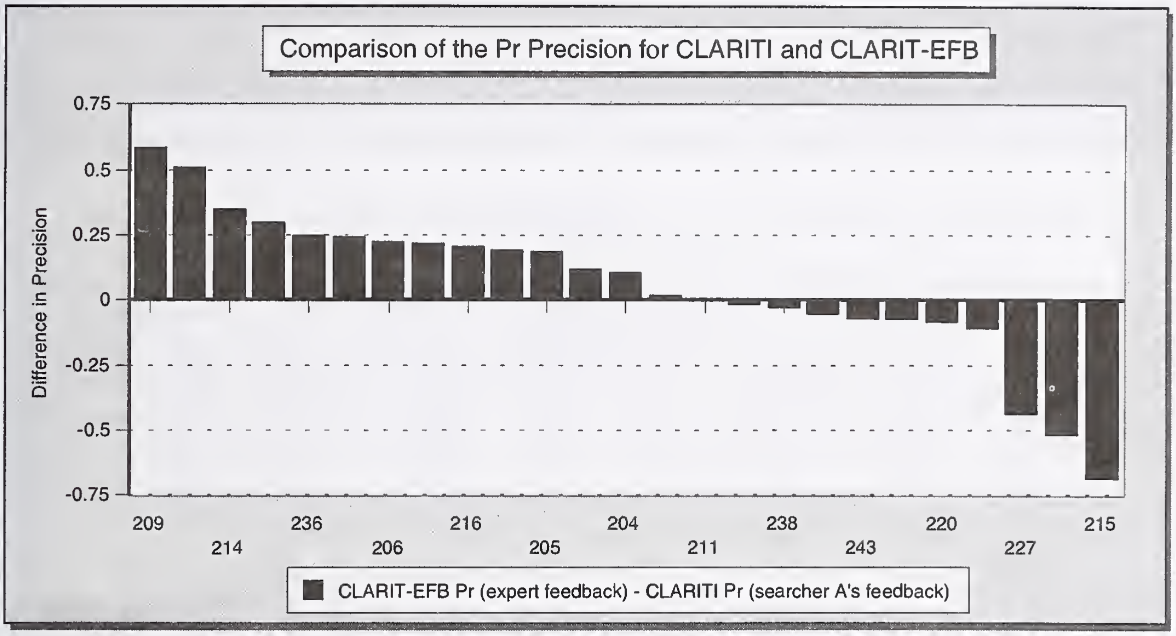


Figure 6. Difference in the P_R precision for the CLARITI and CLARIT-EFB experiments

	CLARITI P_R		**CLARIT-EFB P_R**		**Diff**		**Rel. Diff.**	
	all topics	23 topics	all topics	23 topics	all topics	23 topics	all topics	23 topics
Mean	**0.28**	**0.3**	**0.34**	**0.34**	**0.05**	**0.04**	**217%**	**67%**
Median	**0.19**	**0.22**	**0.3**	**0.3**	**0.1**	**0.02**	**37%**	**34%**
Std Dev	0.22	0.22	0.22	0.23	0.29	0.3	660%	134%
Max	0.77	0.77	0.8	0.8	0.58	0.58	3279%	347%
Min	0.01	0.06	0.08	0.08	-0.69	-0.69	-89%	-89%

Table 8. Summary of the precision statistics

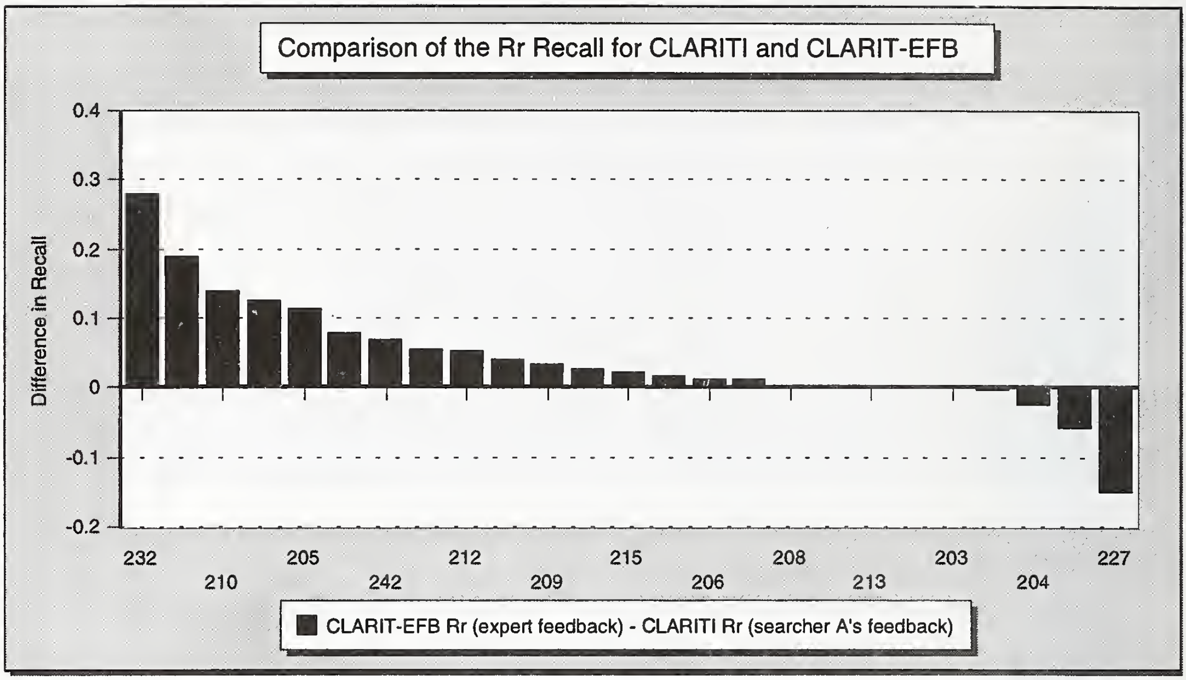


Figure 7. Difference in R_R recall measurements for the CLARITI and CLARIT-EFB experiments

	CLARITI R_R		CLARIT-EFB R_R		Diff		Rel. Diff.	
	all topics	23 topics	all topics	23 topics	all topics	23 topics	all topics	23 topics
Mean	**0.4**	**0.43**	**0.44**	**0.45**	**0.04**	**0.03**	**64%**	**9%**
Median	**0.35**	**0.37**	**0.37**	**0.38**	**0.02**	**0.02**	**7%**	**9%**
Std Dev	0.3	0.29	0.29	0.29	0.08	0.07	240%	43%
Max	1	1	1	1	0.28	0.19	1197%	193%
Min	0.02	0.03	0.03	0.03	-0.15	-0.15	-23%	-23%

Table 9. Summary of the recall statistics

On average, the use of automatic feedback, as applied in the CLARIT-EFB experiment, yields a 4% absolute increase and a 67% relative increase in the precision compared with the CLARITI P_R. The recall improvement amounts to 3% of absolute increase and 9% of relative increase on average. A correct interpretation of these results requires a closer look at the search strategy taken by searcher A in the CLARITI experiment. We note that during the CLARITI search sessions, query augmentation (based either on TREC-4 database or some external database) was used on average only once per topic. Therefore, the improved retrieval performance observed in CLARIT-EFB is a result of both the difference in relevance feedback and the particular search strategy.

4.3 Cross-searcher comparison

In addition to the full set of interactive experiments performed by the searcher A, we have analyzed a set of 13 interactive search sessions performed by searchers B and C. This data enables us to make some preliminary observations on the cross-searcher comparison that will be thoroughly explored in future interactive task experiments.

Tables 10 and 11 show the precision and recall achieved by searchers B and C, respectively.

SEARCHER B					
Query ID	**Tot_Rel**	**Retrieved**	**Rel_ret**	**Prec**	**Recall**
239	123	52	6	**0.17**	**0.07**
242	38	44	6	**0.14**	**0.16**
243	69	96	18	**0.19**	**0.26**
250	86	2	1	**0.5**	**0.01**
Micro Av.	79	49	8.5	**0.25**	**0.13**
Macro Av.				**0.18**	**0.11**

Table 10. Performance statistics for the searcher B

SEARCHER C					
Query ID	Tot_Rel	Retrieved	Rel_ret	Precision	Recall
213	24	4	3	1	0.14
216	5	2	2	1	0.4
215	104	12	9	0.75	0.05
216	36	42	23	0.55	0.64
220	24	4	3	0.75	0.14
227	347	61	44	0.72	0.14
232	9	7	4	0.14	0.14
216	43	17	4	0.14	0.14
238	270	25	14	0.55	0.05
Micro Av.	104	19	11.4	0.63	0.14
Macro Av.				0.6	0.11

Table 11: Performance statistics for the Searcher C

In order to learn more about searching styles of the subjects we collected statistics on:

- time spent per topic
- number of documents viewed per topic
- speed of viewing documents
- number of searches performed per topic
- ratio of viewed and kept documents
- number of terms used in initial queries and added during the search session

computed for each searcher over all the topics that she or he performed as well as over the topics that they 'shared'.

Searcher	Time per Topic	Doc Viewed per Topic	Doc Viewed per min	Searches per Topic	Viewed : Kept Docs	Terms per query first - final	
A All 25 Topics	22	110	5	3	110 : 14	13	33
B	30	98	3	7	98 : 49	8	23
A for B Topics	23	136	6	4	136 : 6	17	36
C	23	98	2	3	48 : 19	9	14
A for C Topics	20	118	6	2	118 : 18	10	32

Table 12.

As is evident from the data in Table 12, searcher A reviewed documents at a much faster pace than the other two searchers: 5-6 documents per minute. Furthermore, this searcher was more conservative in keeping documents as relevant, on average 14 out of 110 reviewed documents. From the query term statistics we note that although searcher A used much larger queries than searcher B in the initial search, the relative increase in number of terms in the final queries was almost the same: the final queries were more than two times larger than starting queries. Searcher C, on the other hand, was much more conservative in adding terms to the query; the increase in the number of query terms for this searcher was only about 50% of the initial query.

It is interesting to observe that searcher B spent a significant amount of time viewing documents for each query, carefully deciding which documents were relevant to the topic. The relatively low precision achieved by this searcher probably reflects a difference in understanding of the topics. Table 5 shows that precision P_V calculated over the viewed and precision P_K over kept documents do not differ significantly for this searcher. This indicates that it was not the searcher's selection of the viewed documents but rather the query formulation that resulted in the low retrieval performance from the expert's point of view (Table 15). It is worth noting that searcher B kept on average 49 documents as relevant.

		A			B	C
		For 25 Topics	For Topics B	For Topics C		
P_V-P_K	Mean	0.12	0.05	0.1	0	0.12
	Median	0.09	0.06	0.14	0	0.15
R_V-R_K	Mean	0.21	0.12	0.35	0.02	0.12
	Median	0.12	0.11	0.42	0.03	0.16

Table 13. Difference in the relevance judgment measured by the absolute difference in recall and precision

		A			B	C
		For 25 Topics	For Topics B	For Topics C		
Precision	Micro	**0.67**	0.77	**0.59**	0.25	**0.63**
	Macro	**0.7**	0.61	**0.71**	0.18	**0.6**
Recall	Micro	**0.11**	0.04	**0.15**	0.13	**0.19**
	Macro	**0.07**	0.04	**0.09**	0.11	**0.11**

Table 14. TREC-4 official precision/recall evaluation

We also found it interesting to compare the searchers on the basis of two additional statistical measures: the USP measure that captures the 'user specific precision' of the system and the LE measure that measures the 'level of effort' involved in identifying new relevant documents. Both of these measurements use the searcher's relevance judgments as the only relevance criteria for document evaluation.

The USP measure for a topic is calculated as the average non-interpolated precision from the searcher's relevance perspective, computed over the viewed documents :

$$USP = Avg_{search}(Avg_{viewed}\frac{\text{\# of kept documents}}{\text{\# of viewed documents}})$$

The LE measure represents the amount of effort required from the user to identify a new relevant document during the search:

$$LE = Avg_{search}(Avg_{viewed\ docs}\frac{\text{\# of (newly) kept documents}}{\text{\# of (newly) viewed documents}})\ .$$

		A			B	C
		For 25 Topics	For Topics B	For Topics C		
USP	Mean	0.16	0.05	0.15	**0.32**	0.24
	Median	0.1	0.05	0.09	**0.34**	0.21
LE	Mean	0.11	0.03	0.08	**0.23**	0.15
	Median	0.05	0.02	0.02	**0.24**	0.09

Table 15. Comparison of the efficiency of the system performance

The measurements presented in the Table 15 show a relatively high 'level of satisfaction' with the system by searcher B both with respect to system precision and the amount of effort needed to identify new relevant documents.

5. CLARIT TREC-4 Secondary Interactive Experiments

In addition to analyzing the results of the primary interactive tasks, we performed an evaluation of the secondary interactive task for CLARITI and CLARIT-EFB. The final queries that were generated during these two experiments were used to retrieve 1000 documents from the TREC-4 database. The corresponding precision/recall curves presented in the Figure 9 show the effects of reliable feedback on the quality of the final query as measured by the achieved retrieval performance.

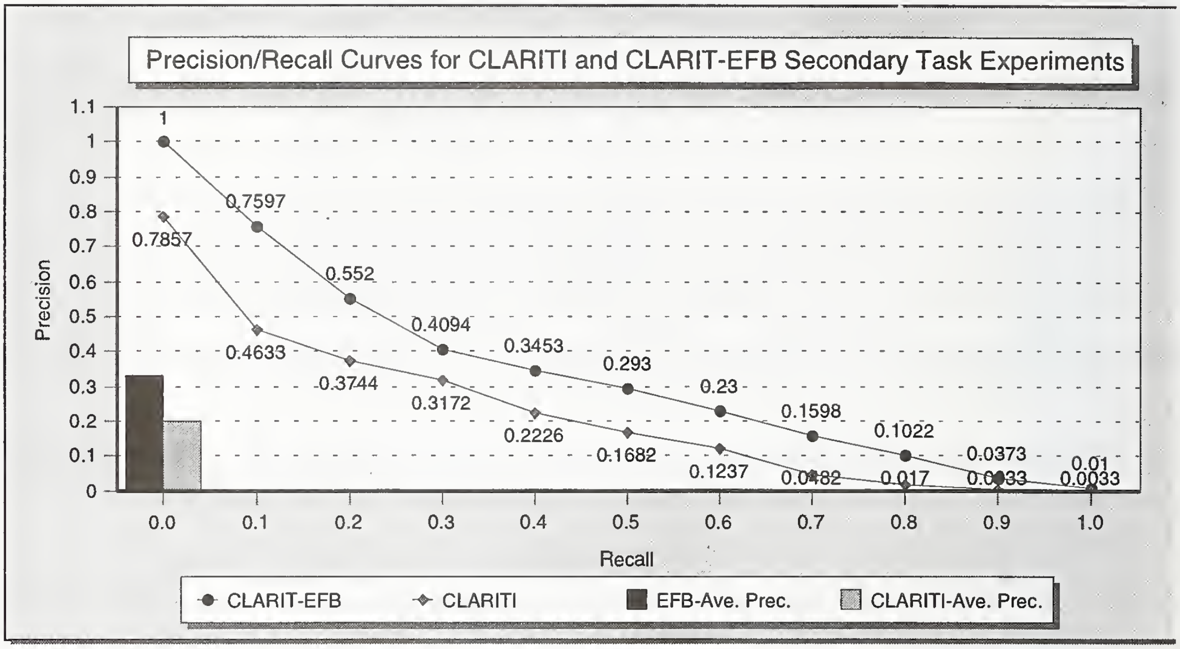


Figure 8. Precision/Recall curves for the CLARITI and CLARIT-EFB Secondary Task Experiments

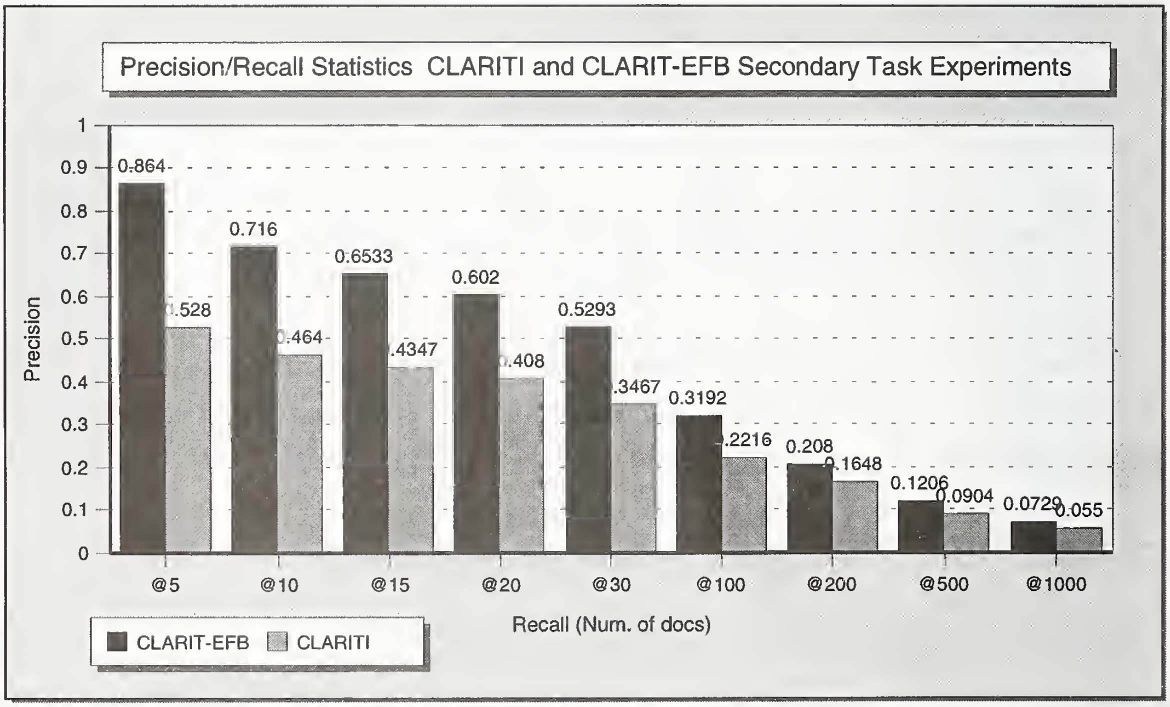


Figure 9. Precision/Recall Statistics for the CLARITI and CLARIT-EFB Secondary Task Experiments

A significant increase in precision and recall for individual queries can be observed from the graphs in Figures 9 and 10 which show the comparison of the CLARITI and CLARIT-EFB runs with the median precision and recall statistics of all TREC-4 secondary interactive task experiments.

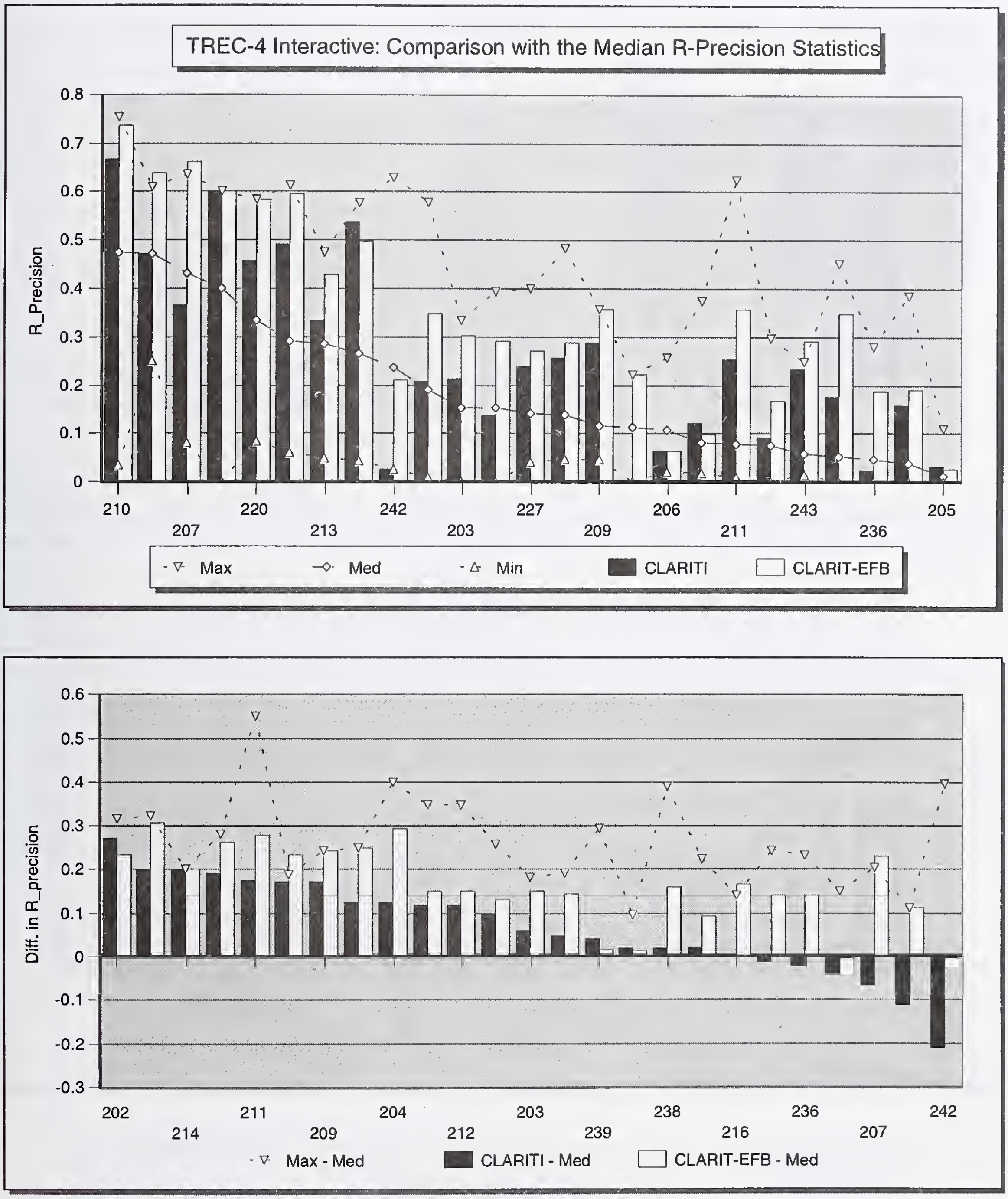


Figure 10. Comparison with the R_Precision statistics of the TREC-4 secondary interactive task

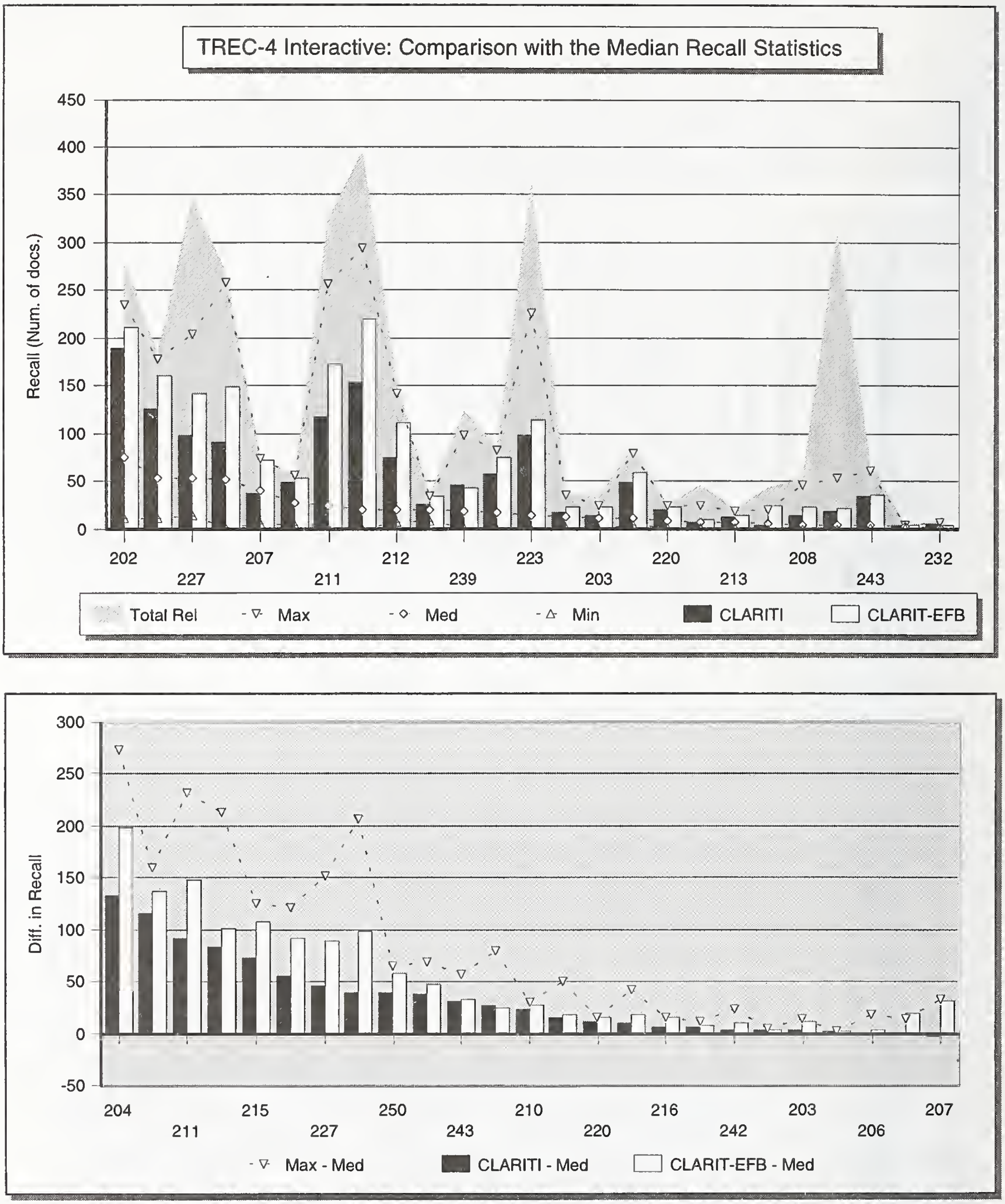


Figure 11. Comparison with the Recall statistics of the TREC-4 secondary interactive task

6. Concluding remarks

Participation in the TREC-4 Interactive task provided us with valuable insights into complex evaluation issues related to fully interactive experiments. In our post-TREC analysis of the CLARIT interactive experiments we concentrated on defining evaluation measures that would be helpful for intra-system evaluation, particularly aimed at identifying and characterizing the important factors in a CLARIT type interactive search. In particular, we define precision/recall measures that help us gradually 'separate' the effects of variables coexisting in the CLARIT TREC-4 Interactive experiments (Table 16). Statistical analysis of these measurements shows the dramatic impact that the user's interpretation of the topic, implicitly reflected in the user's relevance judgment, has on the precision and recall characteristics of the retrieved documents. The 30 minute time constraint on the search session and the relevance feedback have also proven to have significant influence on the retrieval outcome (Table 17).

Coexisting Variables			Eval. Measures	
Doc Selection	**Time Constraint**	**Relevance Feedback**	**Precision**	**Recall**
by the user	yes	by the user	P_K	R_K
by the expert	yes	by the user	P_V	R_V
by the expert	**no**	by the user	P_R	R_R
by the expert	**no**	**by the expert**	P_R	R_R

Table 16.

	Avg. Prec.	Avg. Rel. Improv.		Avg. Recall	Avg. Rel. Improv.
CLARITI P_K	0.12		**CLARITI R_K**	0.11	
CLARITI P_V	0.24	180%	**CLARITI R_R**	0.32	185%
CLARITI P_R	0.28	17%	**CLARITI R_R**	0.4	36%
CLARIT-EFB P_R	0.34	67%	**CLARIT-EFB R_R**	0.44	9%

Table 17.

The significance of reliable user feedback is particularly well demonstrated by the comparison of the two secondary task experiments CLARITI and CLARIT-EFB. Considering the standard TREC-4 precision/recall measures we find an absolute improvement of 13% in average precision and 9% in R-precision. The relative improvements amount to 65% in average precision and 36% in R-precision.

	CLARITI	CLARIT-EFB	Diff	Rel.Diff.
Avg. Prec.	0.2022	0.3336	0.1314	**65%**
R-Prec.	0.2578	0.3503	0.0925	**36%**

Table 18.

While our analysis does include some preliminary inter-searcher comparison, much still remains to be done in that area. In our future research we intend to undertake further development of an appropriate user typology and corresponding evaluation measures.

References

[1] Harman D. (Ed.). (1993). *The First Text REtrieval Conference (TREC-1).* National Institute of Standards and Technology Special Publication 500-207, Gaithersburg, Md. 20899

[2] Harman D. (Ed.). (1994a). *The Second Text REtrieval Conference (TREC-2).* National Institute of Standards and Technology Special Publication 500-215, Gaithersburg, Md. 20899

[3] Harman D. (Ed.). (1995). *The Third Text REtrieval Conference (TREC-3).* National Institute of Standards and Technology Special Publication 500-225, Gaithersburg, Md. 20899

[4] Evans, David A. and Lefferts, Robert G., "Design and Evaluation of the CLARIT-TREC-2 system". In Harman D. (Ed.) *The Second Text REtrieval Conference (TREC-2).* National Institute of Standards and Technology Special Publication 500-215. Washington, DC: Government Printing Office, 1994, 137-150.

Appendix

CLARIT-TREC-4 Interactive System Description

I. System description

1. Screen dump of "typical" screen.	see below
2. Usable features of the interface.	see Section 2 and 3 of the paper
3. Style of interface:	Graphical User Interface

II. Experimental conditions

The CLARITECH team submitted one set of search results for the official evaluation, the results from the CLARITI interactive experiment. All the searches in the experiment were performed by a single searcher.

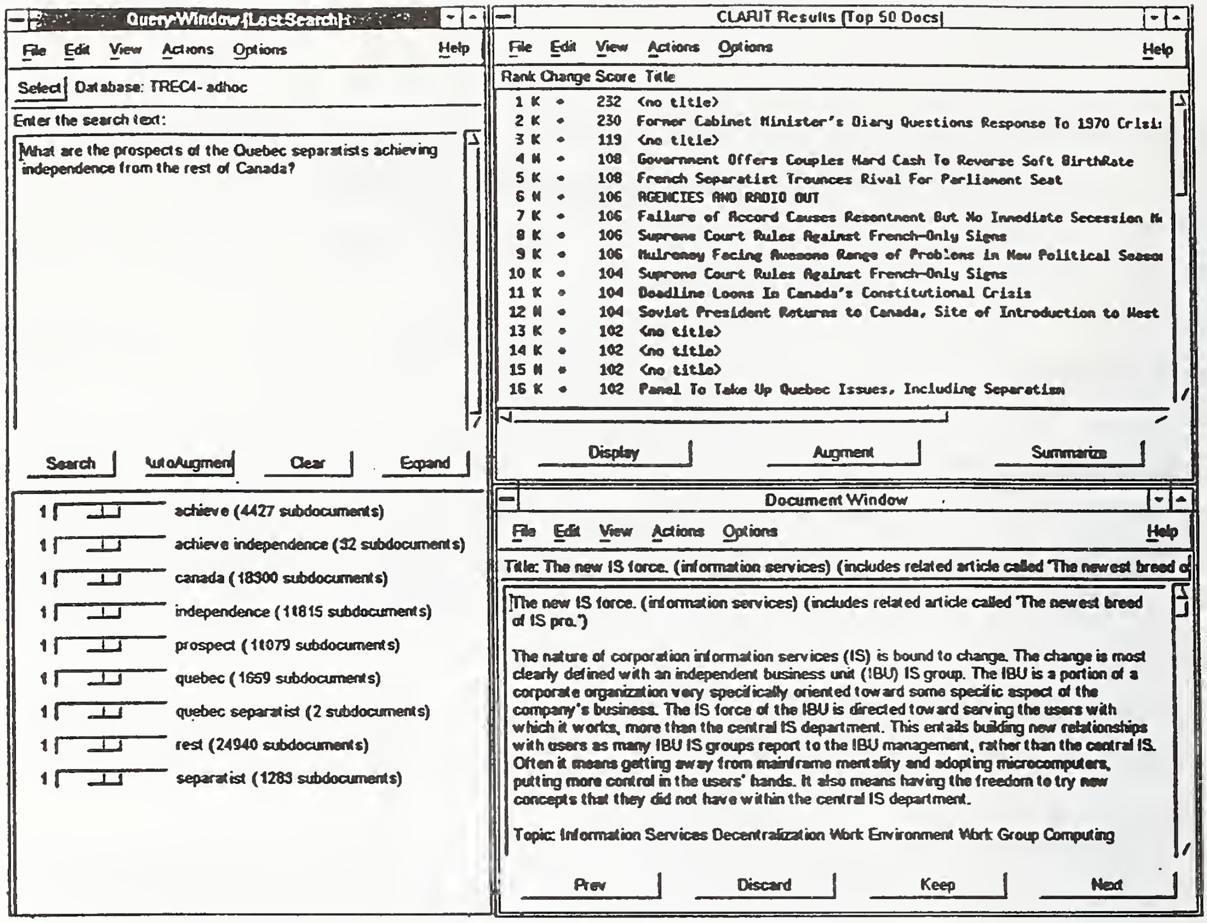
1. Searcher characteristics

a. Number of searchers in experiment:	1
b. Number of searchers per topic:	1
c. Age/age group of searchers.	20-30 years old
d. IR searching experience of searchers.	Intermediate. Searcher is familiar with the CLARIT System but has no experience in professional information searching.
e. Educational level of searchers.	Master's Degree
f. Undergraduate major of searchers.	Linguistics and Philosophy
g. Experience/familiarity with subject of topic.	Searcher was not particularly familiar with any of the topics
h. Work affiliation of searchers.	Employed by CLARITECH Corporation

2. Task description

Here is the description of the TREC-4 Interactive task that was communicated orally to the searcher prior to the experiments:

> Searcher's task is to identify as many relevant documents as possible for each of the 25 TREC topics selected for the Interactive Task. As a starting point searcher can use the query generated by the CLARIT Natural Language Processing of the topic description or can begin the search with his/her own initial formulation of the query. Searching time on any particular topic should not exceed 30 minutes. During that time searcher can formulate as many different queries as he/she desires and can perform as many searches as he/she finds necessary to retrieve relevant documents. All systems' features that are accessible through the interface can be used during the search.
>
> Searcher is advised to read the text of a retrieved documents in order to determine whether the document is relevant to the particular topic. All relevant documents should be 'kept' and non-relevant documents 'discarded'. Upon the completion of the search on a given topic, searcher should save the final query and the final set of kept documents.
>
> Before starting a search session for a new topic, searcher should restart the user interface to ensure that the system will store search information about individual topic's separately.


Query Window [Last Search]
File Edit View Actions Options Help
Select
Search
AutoAugment
Clear
Expand
CLARIT Results [Top 50 Docs]
File Edit View Actions Options Help
Display
Augment
Summarize
Document Window
File Edit View Actions Options Help
Prev
Discard
Keep
Next

Database: TREC4- adhoc

Enter the search text:

What are the prospects of the Quebec separatists achieving independence from the rest of Canada?

1	achieve (4427 subdocuments)
1	achieve independence (32 subdocuments)
1	canada (18300 subdocuments)
1	independence (11815 subdocuments)
1	prospect (11079 subdocuments)
1	quebec (1659 subdocuments)
1	quebec separatist (2 subdocuments)
1	rest (24940 subdocuments)
1	separatist (1283 subdocuments)

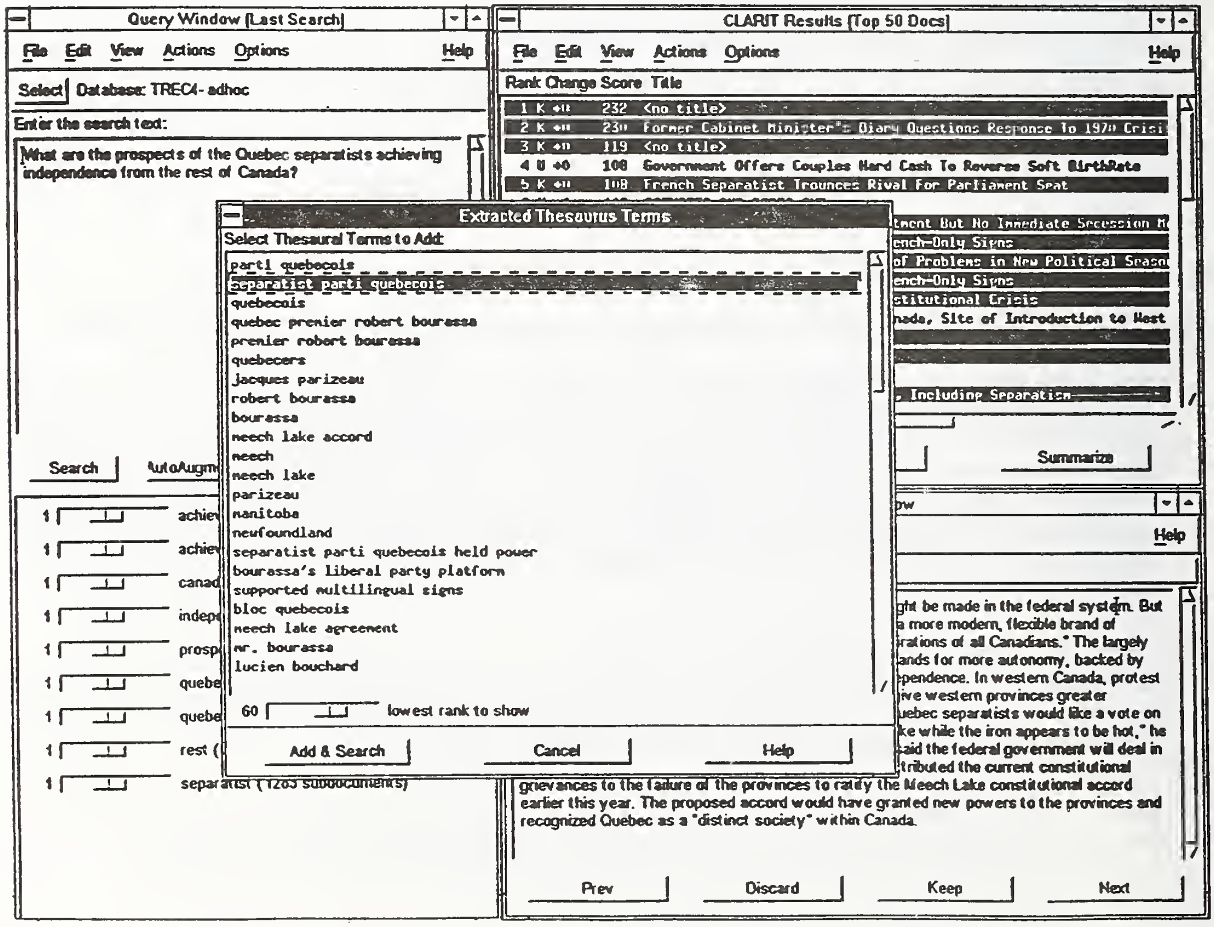
```
Rank Change Score Title
  1 K  •   232  <no title>
  2 K  •   230  Former Cabinet Minister's Diary Questions Response To 1970 Crisi:
  3 K  •   119  <no title>
  4 N  •   108  Government Offers Couples Hard Cash To Reverse Soft BirthRate
  5 K  •   108  French Separatist Trounces Rival For Parliament Seat
  6 N  •   106  AGENCIES AND RADIO OUT
  7 K  •   106  Failure of Accord Causes Resentment But No Immediate Secession M
  8 K  •   106  Supreme Court Rules Against French-Only Signs
  9 K  •   106  Mulroney Facing Awesome Range of Problems in New Political Seaso
 10 K  •   104  Supreme Court Rules Against French-Only Signs
 11 K  •   104  Deadline Looms In Canada's Constitutional Crisis
 12 N  •   104  Soviet President Returns to Canada, Site of Introduction to West
 13 K  •   102  <no title>
 14 K  •   102  <no title>
 15 N  •   102  <no title>
 16 K  •   102  Panel To Take Up Quebec Issues, Including Separatism
```

Title: The new IS force. (information services) (includes related article called 'The newest breed o

The new IS force. (information services) (includes related article called 'The newest breed of IS pro.')

The nature of corporation information services (IS) is bound to change. The change is most clearly defined with an independent business unit (IBU) IS group. The IBU is a portion of a corporate organization very specifically oriented toward some specific aspect of the company's business. The IS force of the IBU is directed toward serving the users with which it works, more than the central IS department. This entails building new relationships with users as many IBU IS groups report to the IBU management, rather than the central IS. Often it means getting away from mainframe mentality and adopting microcomputers, putting more control in the users' hands. It also means having the freedom to try new concepts that they did not have within the central IS department.

Topic: Information Services Decentralization Work Environment Work Group Computing


Query Window [Last Search]
File Edit View Actions Options Help
Select
Search
AutoAugm
CLARIT Results [Top 50 Docs]
File Edit View Actions Options Help
Summarize
Extracted Thesaurus Terms
Add & Search
Cancel
Help
ow
Help
Prev
Discard
Keep
Next

Database: TREC4- adhoc

Enter the search text:

What are the prospects of the Quebec separatists achieving independence from the rest of Canada?

1	achiev
1	achiev
1	canad
1	indepe
1	prosp
1	quebe
1	quebe
1	rest (
1	separatist (1283 subdocuments)

```
Rank Change Score Title
  1 K +n   232  <no title>
  2 K +n   230  Former Cabinet Minister's Diary Questions Response To 1970 Crisi
  3 K +n   119  <no title>
  4 U +0   108  Government Offers Couples Hard Cash To Reverse Soft BirthRate
  5 K +n   108  French Separatist Trounces Rival For Parliament Seat
                tment But No Immediate Secession M
                ench-Only Signs
                of Problems in New Political Seaso
                ench-Only Signs
                stitutional Crisis
                nada, Site of Introduction to West
                , Including Separatism
```

Select Thesaural Terms to Add:

parti quebecois
separatist parti quebecois
quebecois
quebec premier robert bourassa
premier robert bourassa
quebecers
jacques parizeau
robert bourassa
bourassa
meech lake accord
meech
meech lake
parizeau
manitoba
newfoundland
separatist parti quebecois held power
bourassa's liberal party platform
supported multilingual signs
bloc quebecois
meech lake agreement
mr. bourassa
lucien bouchard

60 lowest rank to show

ght be made in the federal system. But a more modern, flexible brand of rations of all Canadians." The largely ands for more autonomy, backed by pendence. In western Canada, protest ive western provinces greater uebec separatists would like a vote on ke while the iron appears to be hot," he said the federal government will deal in tributed the current constitutional grievances to the failure of the provinces to ratify the Meech Lake constitutional accord earlier this year. The proposed accord would have granted new powers to the provinces and recognized Quebec as a "distinct society" within Canada.

3. Training

a. Description of the training process: No system training was necessary since the searcher was already familiar with the CLARIT System and the features in the CLARIT User Interface.

III. Search process

1. Clock time (i.e. real, elapsed time) per search, from the time the searcher was given the topic, until the final query was saved, in seconds.

Mean: 1,320 sec (22 min)
Median: 1,320 sec (22 min)
SD: 300 sec (5 min)
Range: Min: 660 sec (11min) Max: 1,800 sec (30 min)

2. Number of documents "viewed" during the search.

a. Definitions of viewing: Looking at the full text of a document or the document title

b. Number of items viewed per search

Mean: 110
Median: 123
SD: 34
Range: Min: 1 Max: 158

3. Number of iterations per search.

a. Definition of iteration:

'Iteration' is equivalent to query formulation and execution of the search. For each topic we recorded the number of different query formulations submitted for search within the search session.

b. Number of iterations per search.

Mean: 3
Median: 2
SD: 1
Range: Min: 1 Max: 6

4. Number of terms used in queries.

a. Number of terms in the first query of a search, per search.

Mean: 12
Median: 12
SD: 5
Range: Min: 5 Max: 23

b. Number of terms in the final query of a search, per search.

Mean: 33
Median: 26
SD: 21
Range: Min: 7 Max: 96

5. Use of system features.

a. Query augmentation using terminology from the target corpus ('Augment')

Num of times:	Feature invoked	Terms added to the query
Total:	18	17
Mean:	1	1
Median:	1	1
SD:	1	1
Range:	Min: 0 Max: 2	Min: 0 Max: 2

b. Query augmentation using terminology external databases ('Expand')

Num of times:	Feature invoked	Terms added to the query
Total:	13	7
Mean:	1	0
Median:	0	0
SD:	1	1
Range:	Min: 0 Max: 2	Min: 0 Max: 2

6. Number of user errors made per search. N/A

7. Search narrative for topic 236.

In addition to Topic 236, we have performed an analysis of the search session for Topic 206 in order to illustrate the use of more advanced retrieval features of the CLARIT System, such as automatic augmentation of the query with terminology from the target database and use of an external database to identify related and potentially useful terminology automatically .

Analysis of the CLARIT TREC-4 Interactive Search Sessions for the Topics 236 and 206

PART I: Levels of the CLARIT TREC-4 Interactive Search Analysis

1. General Introduction

CLARIT interactive query analysis is done based on both the videotape and the automatically logged files of the interactive experiments. First, rough primitive actions by the subject were extracted from the videotape. Then, more detailed information, such as the exact terms added to the query and the document IDs, was recovered from the logged files including multiple versions of query profiles and search results with marks.

The analysis results are divided into three levels, each of which reveals information about the query at different abstraction levels.

LEVEL1: Primitive events level

This level is the most detailed description of the search in terms of the primitive events defined later. The purpose of the description at this level is to provide a detailed record of the events that occurred in the session, and to serve as the basis for further analysis which may include:

- Time points (and thus time intervals) of keeping documents
- Ratio of relevant, but discarded documents (reflecting subject's relevance judgment ability)

LEVEL2: Summarized events level

This level provides a summarized description of the query session. The event units used at this level of description are generally a summarization of several first level events with some details omitted.

The purpose of this level is to provide a concise, yet informative description of the major events, which reveal the major retrieval techniques used by the subject. Possible analysis at this level may include

- Retrieval techniques/strategies used by the subject
- Contribution of individual techniques to retrieval of new relevant/kept documents.

LEVEL3: Major action level

This level provides a description of the search sessions at the level of CLARIT system functions. This level is a further abstraction of the level 2 description where most of the details of the level 2 are ignored and only information about the usage of CLARIT system functions is kept.

2. Notation

The following is a list of event units used in the three levels of description.

LEVEL1: Primitive events level
<TERM> for query term
<#> for numerical parameter
<db> database parameter
<DOCID> for document ID

1). General
begin-query-session
end-query-session
load-query
save-kept-docs

2). Query-related
view-query
add-term(manual,<TERM), add-term(ext-db,<TERM>),
add-term(fb-thesau,<TERM>)
[where <TERM> is the term added]
change-weight(<TERM>,<#1>, <#2>)
[where <TERM> is the term whose weight is changed;
<#1> and <#2> are the weights of the term before and after
being changed]

3). Doc-related
    view-titles
    view-doc-text(<DOCID>,KEEP)
    view-doc-text(<DOCID>,DICARD)
    view-doc-text(<DOCID>, NOACT)
      [where, <DOCID> is the document id of the viewed doc.]

4). System-action
    submit-query
    parse
    ext-db-expansion(<db>)
    feedback-aug(kept,selected,top(<#>))
      [where <#> is the number of documents used for feedback]
    select-term(<TERM>)
      [where <TERM> is the term selected]

LEVEL2: Summarized events level
<#> for numerical parameter
<db> database parameter

1). General
    begin-query-session
    end-query-session
    load-query
    save-kept-docs

2). Query-related
    view-query
    add-terms(manual,<#>), add-terms(ext-db,<#>), add-terms(fb-thesau,<#>)
      [where <#> is the number of terms added]
    change-weights(<#>)
      [where <#> is the number of terms changed]

3). Doc-related
    view-titles
    view-doc-text(next(<#>),keep(<#>),discard(<#>))
      [where <#> is the number of documents marked]

4). System-action
    submit-query
    parse
    ext-db-expansion(<db>)
    feedback-aug(kept(<#>),selected(<#>),top(<#>))
      [where <#> is the number of documents used for feedback]
    select-terms(<#>)
      [where <#> is the number of terms selected]

LEVEL3: Major action level

1). General
- begin-query-session
- end-query-session
- load-query
- save-kept-docs

2). System-action
- submit-query
- view-and-mark-docs
- change-query-weights
- add-terms-to-query(manual), add-terms-to-query(feedback), add-terms-to-query(ext-db)

PART 2: Video Analysis of Topic 236

1. Primitive Descriptions

Time	*Event*
0.0	begin-query-session
0.0	load-query
0.4	add-term("maritime law",manual)
0.5	add-term("ocean", manual)
0.8	change-weight("current",0)
0.8	change-weight("maritime",2)
0.8	change-weight("maritime law",2)
0.8	change-weight("ocean",2)
1.0	submit-query
1.5	view-doc-text([[7683296]],discard)
2.0	view-doc-text([[5502612]],discard)
	view-doc-text([[26984948]],discard)
	view-doc-text([[8925780]],discard)
	view-doc-text([[19053384]],discard)
	view-doc-text([[2527548]],discard)
	view-doc-text([[9655824]],discard)
	view-doc-text([[18418296]],discard)
	view-doc-text([[1451384]],discard)
	view-doc-text([[18483096]],discard)
	view-doc-text([[27450732]],discard)
	view-doc-text([[18414336]],discard)
	view-doc-text([[24189164]],discard)
	view-doc-text([[2519332]],discard)
	view-doc-text([[27933372]],discard)
	view-doc-text([[7666316]],discard)
	view-doc-text([[8042652]],discard)
	view-doc-text([[9199220]],discard)
3.0	view-doc-text([[7761760]],discard)
	view-doc-text([[8026300]],discard)
	view-doc-text([[7990408]],discard)
	view-doc-text([[7217948]],discard)
	view-doc-text([[7501780]],discard)
	view-doc-text([[7540060]],discard)
	view-doc-text([[7826352]],discard)
	view-doc-text([[7882308]],discard)
	view-doc-text([[7065908]],discard)
	view-doc-text([[8162272]],discard)
	view-doc-text([[8461812]],discard)
	view-doc-text([[7048804]],discard)
4.0	view-doc-text([[7995152]],discard)
	view-doc-text([[34194652]],discard)
	view-doc-text([[8096024]],discard)
	view-doc-text([[7170780]],discard)
	view-doc-text([[7666256]],discard)
	view-doc-text([[7375044]],discard)
	view-doc-text([[8156272]],discard)
	view-doc-text([[10533208]],discard)
	view-doc-text([[8026240]],discard)
	view-doc-text([[7065848]],discard)
	view-doc-text([[8081172]],discard)
	view-doc-text([[7524940]],discard)
	view-doc-text([[32952092]],discard)
	view-doc-text([[2075308]],discard)
	view-doc-text([[4124220]],discard)
	view-doc-text([[7606940]],discard)
	view-doc-text([[7769532]],discard)

```
Time   Event
5.0    view-doc-text([[7882248]],discard)
       view-doc-text([[7134832]],discard)
       view-doc-text([[8162212]],discard)
       view-doc-text([[7769592]],discard)
       view-doc-text([[7394848]],discard)
       view-doc-text([[7082500]],discard)
       view-doc-text([[27563412]],discard)
       view-doc-text([[7606880]],discard)
       view-doc-text([[7666376]],discard)
       view-doc-text([[7450416]],discard)
       view-doc-text([[7438504]],discard)
       view-doc-text([[8765532]],discard)
       view-doc-text([[7785192]],discard)
       view-doc-text([[8146732]],discard)
       view-doc-text([[7497760]],discard)
       view-doc-text([[3684052]],discard)
       view-doc-text([[22500436]],discard)
       view-doc-text([[4162212]],discard)

6.0    view-doc-text([[5723024]],discard)
       view-doc-text([[7943964]],discard)
       view-doc-text([[8017600]],discard)
       view-doc-text([[23213384]],discard)
       view-doc-text([[3590692]],discard)
       view-doc-text([[7218188]],discard)
       view-doc-text([[8099476]],discard)
       view-doc-text([[3657700]],discard)
       view-doc-text([[7334720]],discard)
       view-doc-text([[6435008]],discard)
       view-doc-text([[18924632]],discard)
       view-doc-text([[7364004]],discard)
       view-doc-text([[8695392]],discard)
       view-doc-text([[7138764]],discard)
       view-doc-text([[25351200]],discard)
       view-doc-text([[20111788]],discard)
       view-doc-text([[7979068]],discard)
       view-doc-text([[29485512]],discard)

7.0    view-doc-text([SJMN91-06294205],keep)
       view-doc-text([[20634344]],discard)
       view-doc-text([[29864892]],discard)
       view-doc-text([[21821232]],discard)
       view-doc-text([[20352240]],discard)
       view-doc-text([[7051024]],discard)
       view-doc-text([[7712160]],discard)
       view-doc-text([[7255120]],discard)
       view-doc-text([[7720592]],discard)

8.4    view-doc-text([AP900831-0191],keep)
       view-doc-text([[31975468]],discard)
       view-doc-text([[7769472]],discard)
       view-doc-text([[7809492]],discard)
       view-doc-text([[3694920]],discard)
       view-doc-text([[4549628]],discard)
       view-doc-text([[23680304]],discard)
       view-doc-text([[7998544]],discard)
       view-doc-text([[7542100]],discard)
       view-doc-text([[4139404]],discard)
       view-doc-text([[20329368]],discard)
       view-doc-text([[29042172]],discard)
       view-doc-text([[7855844]],discard)
       view-doc-text([[2805104]],discard)
       view-doc-text([[10239476]],discard)

9.0    view-doc-text([[7828752]],discard)
       view-doc-text([[7703580]],discard)
       view-doc-text([[145100]],discard)
       view-doc-text([[18086284]],discard)
       view-doc-text([[8099084]],discard)
       view-doc-text([[28199772]],discard)
       view-doc-text([[7242068]],discard)
       view-doc-text([[36053300]],discard)
       view-doc-text([[8004844]],discard)
       view-doc-text([[7182512]],discard)
       view-doc-text([[7855724]],discard)
       view-doc-text([[8015348]],discard)
       view-doc-text([[1610336]],discard)

10.0   view-doc-text([AP880725-0053],keep)
       view-doc-text([[3675276]],discard)
       view-doc-text([[8439184]],discard)
       view-doc-text([[8961844]],discard)
       view-doc-text([[7492688]],discard)
       view-doc-text([[21648]],discard)
       view-doc-text([[3683876]],discard)
       view-doc-text([[7379844]],discard)
       view-doc-text([[10812884]],discard)
       view-doc-text([[2109364]],discard)
       view-doc-text([[7397548]],discard)
       view-doc-text([[7388248]],discard)
       view-doc-text([[7199464]],discard)
```

```
Time   Event
       view-doc-text([[7318276]],discard)
       view-doc-text([[7096516]],discard)
       view-doc-text([[7641352]],discard)
       view-doc-text([[29043792]],discard)
       view-doc-text([[18245620]],discard)
       view-doc-text([[21668644]],discard)
       view-doc-text([[21710720]],discard)
       view-doc-text([[5647544]],discard)
       view-doc-text([[22257588]],discard)
       view-doc-text([[7955304]],discard)
       view-doc-text([[17410508]],discard)
       view-doc-text([[7973308]],discard)
       view-doc-text([[7347860]],discard)
       view-doc-text([[7980328]],discard)
       view-doc-text([[22457500]],discard)
       view-doc-text([[7505260]],discard)
```

```
Time   Event
       view-doc-text([[27410112]],discard)
10.8    add-term("admiralty law",manual)
10.9    change-weight("admiralty", 0)
11.0    change-weight("admirality law",2)
11.2    submit-query
11.5    view-titles
13.0    save-kept-docs
13.0    end-query-session
```

Analysis:
  Time points for keeping documents
     At 7.0 minutes 1st keep
     At 8.4 minutes 2nd keep
     At 10.0 minutes 3rd keep

  START -7.0->[keep1]->1.4->[keep2]->1.6->[keep3]

2. Summarized Description

```
Time   Summarized Events
0.0    begin-query-session
0.0    load-query
0.4    add-terms(["maritime law","ocean"],manual)
0.8    change-weights([("current",0),("maritime",2),
                 ("maritime law",2), ("ocean",2)])
1.0    submit-query
1.5    view-doc-text(discard(83))
7.0    view-doc-text(keep(1))
7.1    view-doc-text(next(8))
8.4    view-doc-text(keep(1))
8.5    view-doc-text(next(44))
10.0    view-doc-text(keep(1))
10.1    view-doc-text(next(13))
10.8    add-terms(["admiralty law"],manual)
10.9    change-weights([("admiralty", 0),("admirality law",2)])
11.2    submit-query
11.5    view-titles
13.0    save-kept-docs
13.0    end-query-session
```

Analysis:

1). Major techniques used:

Add terms manually
Change query term weights

2). Time quota analysis

0.5 minutes for adding terms
0.5 minutes for changing weights
9.2 minutes for viewing and marking docs
1.5 minutes for viewing titles
1.3 minutes used by systems
Total time: 13 minutes

3. Major Action Description

Time	*Major Actions*
0.0	begin-query-session
0.0	load-query
0.4	add-terms-to-query(manual)
0.8	change-query-weights
1.0	submit-query
1.5	view-and-mark-docs
10.8	add-terms-to-query(manual)
10.9	change-query-weights
11.2	submit-query
11.5	view-titles
13.0	save-kept-docs
13.0	end-query-session

Analysis:

CLARIT functions illustrated: (from the analysis)

Basic functions:

1. load a query
2. parse and submit a query
3. view and mark documents
4. view document titles
5. save-kept-docs

Advanced functions:

1. manual addition of query terms
2. change weights of query terms

PART 3 Video Analysis of Topic 206

Search session of the Topic 206 includes two CLARIT System features that were not used during the search on the Topic 236:

- external database query expansion
- feedback thesaurus extraction

We include the analysis of the search session for the Topic 206 to illustrate how these two features are used by the searcher.

Time	*Event*	*Major event*
0.0	begin-query-session	begin-query-session
0.0	load-query	load-query
0.2	add-terms(manual,?)[3]	add-terms-to-query(manual)

Time	*Event*	*Major event*
0.3	parse	
0.6	change-weights(?)	change-query-weights
1.0	submit-query	submit-query
1.1	view-title	view-and-mark-docs
1.3	view-doc-text()	
1.5	view-doc-text(keep(3))	
1.8	view-doc-text()	
1.9	view-doc-text(keep(1))	
2.0	view-doc-text(keep(1))	
2.1	view-doc-text()	
2.2	view-doc-text(keep(1))	
2.3	view-doc-text()	
2.6	view-title	
2.8	view-doc-text()	
3.0	view-doc-text()	
3.2	view-title	
3.6	view-title	
4.0	view-doc-text()	
4.1	view-title	
4.5	view-doc-text()	
5.0	view-title	
5.5.	view-doc-text()	
5.6	view-title	
5.7	view-doc-text()	
5.9	view-title	
6.0	view-doc-text()	
6.1	view-title	
6.3	view-doc-text()	
6.4	view-title	
6.5	view-doc-text()	
6.6	view-title	
6.7	view-doc-text()	
6.9	view-title	
7.0	{view-title, view-doc-text()}+	
9.8	**feedback-aug(kept(?))**	**add-terms-to-query(feedback)**
10.2	select-terms(7)	
10.3	add-terms(7)	
10.5	submit-query	
10.9	view-title	
11.0	**ext-db-expansion(?)**	**add-terms-to-query(ext-db)**
11.2	select-terms(0)	(no terms added)
11.4	**ext-db-expansion(?)**	
11.6	select-terms(0)	
12.0	save-kept-docs	save-kept-docs
12.0	end-query-session	end-query-session

[3] We use a question mark to indicate uncertainty about the user's action because of the difficulty in analyzing the video tape recording of the search session.

Acquaintance:
Language-Independent Document Categorization by N-Grams

Stephen Huffman
Department of Defense
Ft. George G. Meade, MD 20755-6000

Acquaintance is the name of a novel vector-space n-gram technique for categorizing documents. The technique is completely language-independent, highly garble-resistant, and computationally simple. An unoptimized version of the algorithm was used to process the TREC database in a very short time.

Acquaintance is the name of a technique for information processing that combines the robustness of an n-gram-based algorithm with a novel vector-space model. Acquaintance gauges similarity among documents on the basis of common features, permitting document categorization based on a common language, a common topic, or common subtopics. The algorithm is completely language- and topic- independent, and is resistant to garbling even at the 10% to 15% (character) level. Acquaintance is fully described in Damashek, 1995. The TREC-3 conference provided the first public demonstration and evaluation of this new technique, and TREC-4 provided an opportunity to test its usefulness on several types of text retrieval tasks.

The Acquaintance algorithm can be used for processing sets of documents in two distinct ways. One method explores the conceptual space of a set of documents by determining the degree of similarity among all the documents in that set. When the documents are then viewed with a visualization tool that arranges them so that the distance between them corresponds with their putative degree of similarity, the conceptual space defined by those documents becomes apparent. That is, those documents which are similar, and thus most probably related by language or topic, will cluster together. Furthermore, documents that relate to several different topics will be obvious due to their positions and the strengths of their connections to more than one cluster of documents. Those documents which are not clearly similar to any others in the set will stand alone and unconnected to other documents. This mode of using Acquaintance is very useful when exploring the contents of a large and unknown database, and was used very successfully when applied to the interactive task at TREC-4.

Acquaintance can also be used for the more traditional task of retrieving documents from a database based on specific queries. When used in this manner, reference documents are compared to the documents in the database. Those documents in the database which are similar to the reference documents can be quickly identified. Using Acquaintance in this fashion most closely approximates many of the tasks in TREC, and variations on this latter method were used to process most of the data in TREC-4.

Methodology

N-Gram Processing

The Acquaintance algorithm begins by processing texts in a manner very similar to traditional n-gram based techniques. An n-wide window is stepped through text, moving one character at a time. From each n-gram lying within the window, a hash function generates a value that is treated as an address in a document vector, and the contents of that vector address are incremented by one. When all of the n-grams in the document have been processed, the document vector is normalized by dividing the frequency count of n-grams at each vector address by the total number of n-grams in the document. Thus, the sum of the normalized counts of the n-grams in the document vector will sum to one.

Centroid Subtraction

A crucial aspect of Acquaintance when gauging similarity among documents is the subtraction of a centroid vector from the document vectors. The centroid in Acquaintance defines a context within which a set of documents can be usefully compared. This method of subtracting a centroid stands in contrast to more traditional vector-space models which frequently use some form of multiplicative weighting, which results in a rescaling of the axes in the vector space.

The centroid vector characterizes those features of a set of documents that are more or less common to all the documents, and are therefore of little use in distinguishing among the documents. The Acquaintance centroid thus automatically captures, and mitigates the effect of, those frequent but generally undiagnostic features of the language that are traditionally contained in stop lists and removed by stemming algorithms.

The creation of the centroid vector for a set of documents is straightforward and language independent. After each separate document vector is created, the normalized frequency for each n-gram in that document is added to the corresponding address in a centroid vector. When all documents have been processed, the centroid vector is normalized by dividing the contents of each vector address by the number of documents that the centroid characterizes. A centroid thus represents the "center of mass" of all the document vectors in the set.

Computing Similarity Scores

Once documents are characterized by normalized document vectors, the resulting vector-space model permits the use of geometric techniques to gauge similarity among the documents. When comparing a set of document vectors to a set of reference vectors, the cosine of the angle between each document vector and each reference vector, as viewed from the centroid, is computed using Equation 1:

$$S_{mn} = \frac{\sum_{j=1}^{J}(x_{mj}-\mu_j)(y_{nj}-\mu_j)}{\left[\sum_{j=1}^{J}(x_{mj}-\mu_j)^2\sum_{j=1}^{J}(y_{nj}-\mu_j)^2\right]^{1/2}} = \cos\theta_{mn}, \qquad m,=1,\ldots,M, \quad n=1,\ldots N \qquad (1)$$

where the vectors x_m, $m \in 1,\ldots,M$ are the M document vectors, the vectors y_n, $n \in 1,\ldots,N$ are the N reference vectors in a J-dimensional space, and μ is the centroid vector.

A cosine value of 1.0 indicates that the document and reference vectors are perfectly correlated (or identical), a value of minus 1.0 that they are perfectly anticorrelated (or antithetical), and a measure of 0.0, that they are uncorrelated (or orthogonal). A great deal of experimentation has been done using this scoring method for gauging topic similarity, and a clear idea of how the measure behaves as features such as n-gram length and garbling are varied (Huffman, in process) has been obtained.

Acquaintance at TREC-4

System Parameters and Text Processing Procedures

In TREC-3, Acquaintance participated for the first time, and was used in just the routing and adhoc tasks. The purpose of participation in TREC-3 was to get a feel for how well a purely statistical system would work compared to more linguistically sophisticated systems. In TREC-4, Acquaintance participated in a much broader range of tasks, including the routing, ad hoc, interactive, filtering, confusion, and Spanish tracks. While the details of the individual tracks will be discussed below, the same software and basic procedure were used in each track.

For the work in TREC-4, a generic, unoptimized version of Acquaintance, written in ANSI C, was used. The TREC data was processed on a heavily time-shared Cray YMP. Both the routing and ad hoc tasks were run as overnight background jobs, and each took less than 8 hours clock time to finish. For most tasks, the n-gram length was five, and the document vector length (or hash table length) was 262144. The only occasions where the n-gram length differed from five was while processing the twenty percent garbled data for the confusion track, when four-grams were used, and for two of the filtering runs, for which seven-grams were used.

Acquaintance requires almost no preprocessing of the documents. To prepare the TREC database, the SGML tags and headers were stripped from the data, and only characters between the TEXT tags were processed. Acquaintance ignored all non-alphabetic characters in the text and translated all lowercase alphabetic characters to uppercase characters.

Routing

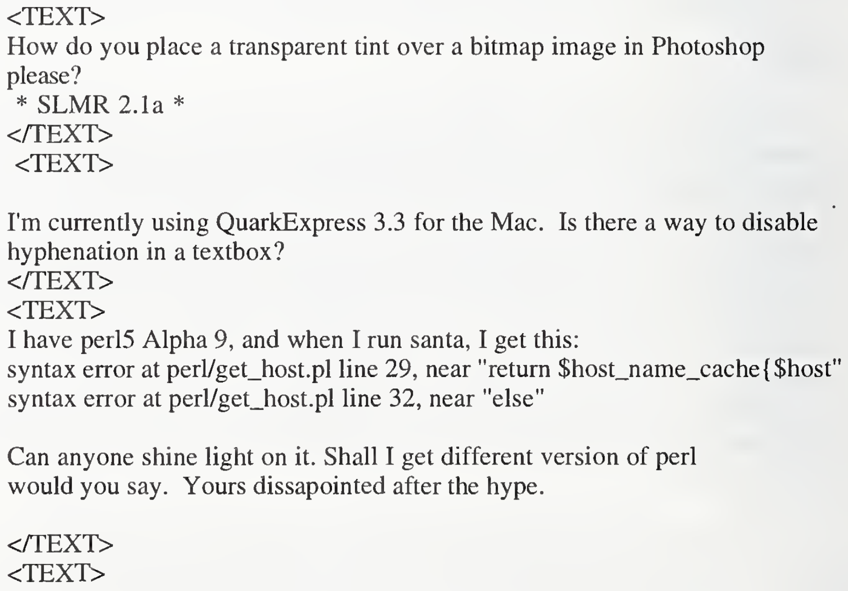
The routing task in TREC simulates the process of filtering an incoming stream of documents according to predefined criteria. Participants are given the topic descriptions (which are taken from previous year's TREC conferences) early in the year. However, the the database of documents is not made available until the queries created from the topic descriptions have been formulated and sent into NIST. In addition, and more importantly for Acquaintance, the list of those documents which were judged relevant to each topic is made available to participants. Thus, a large corpus of potential reference documents is available for each routing topic. However, there is no guarantee that the relevant documents from previous years will in fact be representative of the set of documents used as the database in the current year. In TREC-3, the documents used for reference and for the database were very similar. In TREC-4 they were not, and that fact caused problems for Acquaintance.

To perform the routing task, the AP newswire documents from TREC-3 which were defined to be relevant to each of the routing topics were recovered. The goal was to find a useful subset of those documents to use as reference documents against which to compare the documents in the database. To accomplish this, all the supposedly relevant documents for a particular topic were scored against each other, using the Acquaintance metric. Then, that set of documents and associated scores were submitted to the Parentage tool. One feature of this tool, which will be described more fully below, applies graph theory to sets of scored documents to determine which documents in that set are the most highly connected. Taking advantage of this feature, roughly the 50 most highly connected documents for each topic were selected. Those documents constituted the final set of reference documents for that topic. This process thus produced a set of about 2500 reference documents against which the documents in the database were measured for similarity.

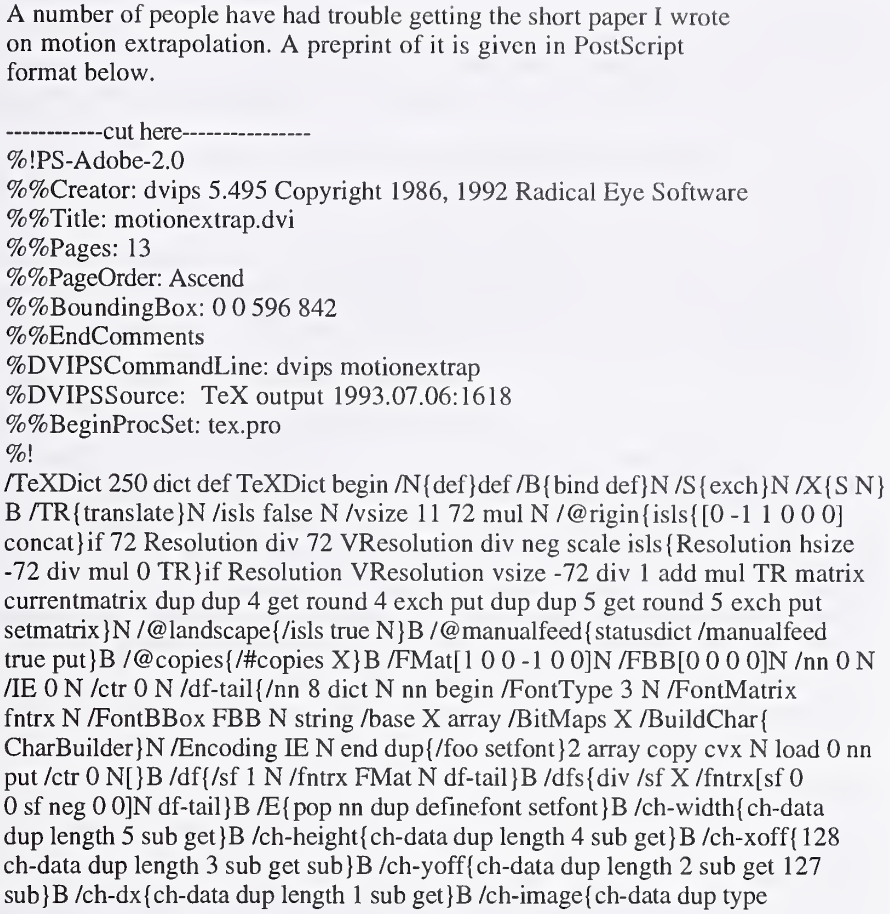
To find relevant documents in the database, a document vector from each document was created and the cosine of the angle between that document vector and each of the reference vectors from each topic was computed, according to Eq. (1). If a document scored above 0.25 when compared to a reference vector, that document's number and score were stored, along with which topic it scored well against. After all documents in the database were compared to all reference vectors, the documents were sorted by topic and score, duplicate documents within topics were removed, and a ranked list of documents gauged similar to at least one reference document in each topic was created.

One serious problem on this task was that the language and style of the reference documents was frequently quite different than that of the documents in the database. The reference documents were in large part drawn from newswire stories that presented a page or so of text discussing a single topic in some detail. The database, in contrast, was weighted towards documents with very different style and content. These documents included Federal Register documents, which tend to be quite large and generally quite diverse in topic and diffuse in style, as well as quite a bit of data from newsgroups, in which language was also quite unlike that of the reference documents.

The newswire documents were particularly difficult for Acquaintance to deal with. An example of some fairly typical texts in the newsgroups are shown in Figure 1 (names and addresses in the body of the text have been removed). The TEXT SGML tags separate the different messages.



```
<TEXT>
How do you place a transparent tint over a bitmap image in Photoshop
please?
 * SLMR 2.1a *
</TEXT>
 <TEXT>

I'm currently using QuarkExpress 3.3 for the Mac.  Is there a way to disable
hyphenation in a textbox?
</TEXT>
<TEXT>
I have perl5 Alpha 9, and when I run santa, I get this:
syntax error at perl/get_host.pl line 29, near "return $host_name_cache{$host"
syntax error at perl/get_host.pl line 32, near "else"

Can anyone shine light on it. Shall I get different version of perl
would you say.  Yours dissapointed after the hype.

</TEXT>
<TEXT>
```



A number of people have had trouble getting the short paper I wrote on motion extrapolation. A preprint of it is given in PostScript format below.

```
------------cut here----------------
%!PS-Adobe-2.0
%%Creator: dvips 5.495 Copyright 1986, 1992 Radical Eye Software
%%Title: motionextrap.dvi
%%Pages: 13
%%PageOrder: Ascend
%%BoundingBox: 0 0 596 842
%%EndComments
%DVIPSCommandLine: dvips motionextrap
%DVIPSSource:  TeX output 1993.07.06:1618
%%BeginProcSet: tex.pro
%!
/TeXDict 250 dict def TeXDict begin /N{def}def /B{bind def}N /S{exch}N /X{S N}
B /TR{translate}N /isls false N /vsize 11 72 mul N /@rigin{isls{[0 -1 1 0 0 0]
concat}if 72 Resolution div 72 VResolution div neg scale isls{Resolution hsize
-72 div mul 0 TR}if Resolution VResolution vsize -72 div 1 add mul TR matrix
currentmatrix dup dup 4 get round 4 exch put dup dup 5 get round 5 exch put
setmatrix}N /@landscape{/isls true N}B /@manualfeed{statusdict /manualfeed
true put}B /@copies{/#copies X}B /FMat[1 0 0 -1 0 0]N /FBB[0 0 0 0]N /nn 0 N
/IE 0 N /ctr 0 N /df-tail{/nn 8 dict N nn begin /FontType 3 N /FontMatrix
fntrx N /FontBBox FBB N string /base X array /BitMaps X /BuildChar{
CharBuilder}N /Encoding IE N end dup{/foo setfont}2 array copy cvx N load 0 nn
put /ctr 0 N[}B /df{/sf 1 N /fntrx FMat N df-tail}B /dfs{div /sf X /fntrx[sf 0
0 sf neg 0 0]N df-tail}B /E{pop nn dup definefont setfont}B /ch-width{ch-data
dup length 5 sub get}B /ch-height{ch-data dup length 4 sub get}B /ch-xoff{128
ch-data dup length 3 sub get sub}B /ch-yoff{ch-data dup length 2 sub get 127
sub}B /ch-dx{ch-data dup length 1 sub get}B /ch-image{ch-data dup type
```

Figure 1. Examples of texts from data for routing task

One problem was that many of the documents were so short that it was difficult to create a good statistical profile of them. Furthermore, the very unusual formats of some documents, as shown by the last example above, helped muddle the statistics on some files of documents. Paradoxically, had most or all the documents been in say, PostScript format, the system would have been better able to group them on the basis of content, as the PostScript "background" would have been accounted for and removed by the statistic profile created by the centroid. In any case, the content and style of these messages was very different from the newswire documents that characterized most of the reference documents.

In a effort to lessen the problems caused by the very different styles of language used in the reference documents and the documents from the database, two centroid vectors were used instead of one. First, a reference centroid vector from all of the reference documents was created. Then, documents from the database were read in one file at a time, and a centroid vector for that set of documents was created to capture the commonality among them. When comparing a document vector to a reference vector, the appropriate centroid was subtracted from the corresponding vectors, as shown in Equation 2:

$$S_{mn} = \frac{\sum_{j=1}^{J}(x_{mj}-\mu_j)(y_{nj}-v_j)}{\left[\sum_{j=1}^{J}(x_{mj}-\mu_j)^2\sum_{j=1}^{J}(y_{nj}-v_j)^2\right]^{1/2}} = \cos\theta_{mn}, \quad m=1,\ldots,M, \; n=1,\ldots N \qquad (2)$$

where the vectors x_m, $m \in 1,\ldots,M$ are the M document vectors, the vectors y_n, $n \in 1,\ldots,N$ are the N reference vectors in a J-dimensional space, μ is the centroid vector for the current file of documents from the database, and v is the centroid for the set of reference documents.

The performance of the Acquaintance system on the routing track was rather poor. In fact, it performed significantly worse in TREC-4 than it did on the same track in TREC-3. In terms of average precision, it scored above the median only three times out of fifty. The reason for this was that in TREC-4 there was a much greater degree of mismatch between the documents that were used as references and the documents that were in the database. Since Acquaintance is a purely statistical system, if the statistics of the reference documents are significantly different from the documents in the database, it cannot perform well. In a real-world situation, if performance were this poor, one would add samples of documents whose content and style more closely modeled those in the database to the set of reference documents. The reference documents that were used for this task would be used only as a first approximation, and a set of more useful reference documents would either supplement or replace the original references.

Ad Hoc

The ad hoc task simulates the activity of a user who submits queries to a static database. The database is made available for the participants to train on early in the year, while the topic descriptions are only made available for a short time before the results of searches based on those descriptions are to be submitted.

In previous years the topic descriptions for the ad hoc task were fairly detailed. The topics consisted of a paragraph or two describing the topic, along with guidance as to what was and was not considered relevant to that topic, as well as a list of what amounted to keywords that helped define the topic even further. This year, the topics were very terse; in fact, some were almost telegraphic. For instance, topic 202 read “Status of nuclear proliferation treaties -- violations and monitoring.” On the other hand, some were more wordy, but actually much less specific, such as topic 216, “What research is ongoing to reduce the effects of osteoporosis in existing patients as well as to prevent the disease occurring in those unaffected at this time.” Logically, this topic boils down to “research on osteoporosis;” all other terms are redundant or uninformative. These extremely short topic descriptions are not untypical of spontaneous user queries, but by themselves they are not long enough from which to generate very solid statistics.

Due to the very sparse nature of this year’s queries, query generation was performed manually for all the ad hoc-based tasks. Since Acquaintance is a statistically-based algorithm, some minimum amount of vocabulary pertaining to the topic must be available for the system to reliably select documents with similar statistical profiles from a database. A few (usually no more than 5 or 6) words or phrases were therefore manually added to the supplied query, using the general subject knowledge of the users (Marc Damashek and Steve Huffman). That process took only a minute or two for each query. In addition, some terms deemed uninformative were removed. As an example, topic 201 originally read "What procedures should be implemented to ensure that proper care is given to children placed under the au pair's responsibility." This was changed this to read "au pair, children, proper care, nanny, nannies, caregiving, au pair, caretaker."

At this point, the modified queries were run against the documents in the database, and the highest scoring documents were returned. Those documents were then scored against each other. The 50 or so documents that were most highly connected to the other documents in the set, as determined by the Parentage tool, were automatically selected. These documents were then used as the reference documents for the final phase of scoring. If a document from the database scored above 0.25 when compared to the reference vector, that document's number and score were stored. Finally the documents were sorted by score, duplicate documents in each topic were removed, and a ranked list of documents gauged similar to at least one reference document was created.

The results on the ad hoc task in TREC-4 were considerably better than those in TREC-3. In spite of the sparseness of the queries, Acquaintance performed moderately well, scoring above the median in average precision on 15 out of the 49 topics. It would seem that the technique of running a first pass through the data to choose good candidate documents, and then using the most highly connected of those as the final set of reference documents, was more effective than last year's strategy of just using the given topic as the reference document.

Interactive

The interactive task permits the user of a system to interact with that system in a more natural fashion than the ad hoc task. The user is not limited to submitting a single query and simply accepting what the system returns. Rather, the user can examine the system's response to a query, and use that information to choose relevant documents, and/or further refine the query. The queries for this task were the a subset of those used for the ad hoc task.

There were actually two possible tasks for participants in this track. The first was simply to retrieve relevant documents, as in the basic ad hoc task. The second task was to use the system to create a new query, and submit the documents retrieved based on that query. The Acquaintance algorithm performed the first of these two tasks.

For this task, a somewhat different method was attempted than that used by most participants. A tool was used that shows the user the entire universe of documents that might be related to the topic at hand, and permits the user to roam through that universe, examining and/or selecting whole clusters of topic-related documents at one time. This is in contrast to those systems in which the user examines some set of documents returned by a system for a query, and then refines or resubmits the query based on the content of that set of documents.

This was accomplished with the Parentage information visualization system created by Dr. Jonathan Cohen (Cohen, 1995). For each topic, the 1000 top-scoring documents were found using the same procedure as the basic ad hoc task. Those 1000 documents were then scored against each other using the Acquaintance algorithm. Finally the documents and scores were submitted to the Parentage system, which graphed the relationships among the documents in that set.

In addition to visually mapping how documents cluster together, and how documents and clusters of documents relate to each other, the Parentage tool can automatically label each cluster of documents with a set of terms which characterize those words and phrases which cause that cluster of documents both to stand out from the rest, and pull together the documents within the cluster. These terms are referred to as "highlights." Parentage does this by using a modified version of the Acquaintance algorithm, using n-gram statistics and a form of centroid subtraction. An example of this can be seen in figure 2. This figure shows a screen shot of a small part of the Parentage graph for the documents from topic 242.

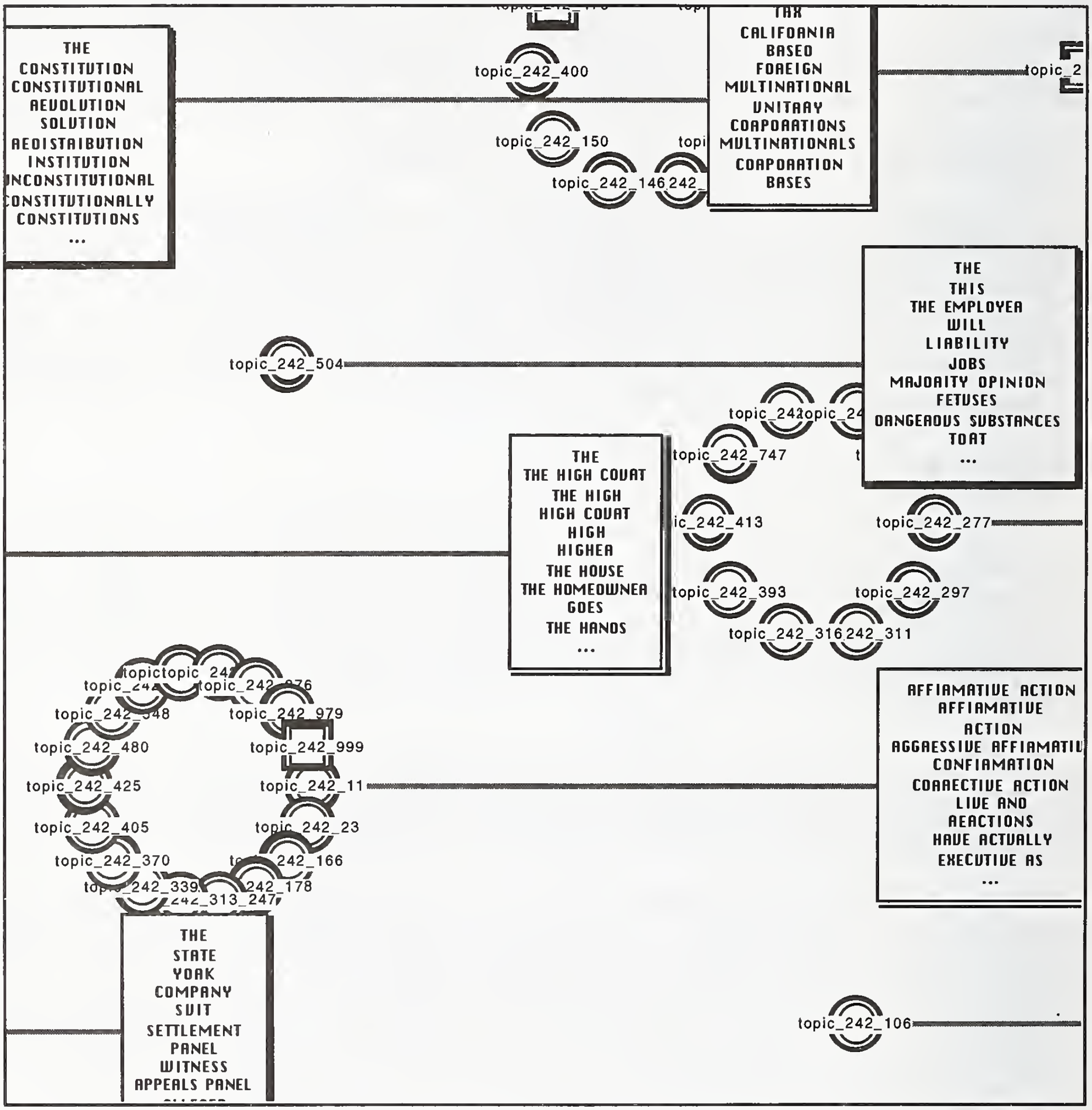


Figure 2. Portion of a Parentage display of a set of documents with automatically generated labels.

It can be seen in Figure 2 that each cluster of documents is shown with a list of highlights. Rather than needing to roam through the whole information space, the user can search for specific terms in either the highlights lists, or in the text of the documents themselves. This will put the user directly onto clusters of documents that may be of interest. Alternatively, if there is a good exemplar document for a topic, one can go directly to that document, and follow the paths of relationships leading from that.

In Figure 2, the user was looking for the term "affirmative action," since that was part of the subject of topic 242, which reads "How has affirmative action affected the construction industry?" By typing in the keyword "affirmative action," the user was moved directly to a cluster of documents dealing with that topic. The user at this point could continue searching for other clusters of documents, perhaps with other keywords, such as "construction." Once the potentially useful clusters of documents were isolated, the user can examine the documents individually, or merely select entire groups of documents. In cases where the highlights are suggestive but not diagnostic, the user can actually read all the documents in a cluster (all together or one at a time) in a window on the screen, and if the documents appear relevant, an entire cluster can be selected.

Another powerful feature of the Parentage tool is that the user can re-cluster sets of documents within their own context. For example, the cluster of documents on affirmative action can be moved to its own window, and the documents can be re-clustered and re-labeled based just on the text of the documents within that particular cluster. This effectively removes the common elements from the documents (in this case presumably terms dealing with affirmative action) leaving subtopics as the basis for clustering and labeling. Thus, if a user wanted a set of documents on a broad topic, such as affirmative action, the cluster so labeled in Figure 2 could be chosen with a fairly high degree of confidence that all the documents in it would be relevant to that topic. On the other hand, if the user wanted to find documents dealing with affirmative action in a certain context, Parentage could be used to examine subtopics within the overall cluster of documents on affirmative action.

It should be clear that whole clusters of related, relevant documents can be located and selected from the conceptual map in a remarkably short time. For any given topic, the users (again, Marc Damashek and Steve Huffman) needed to spend an average of less than ten minutes per topic finding the relevant documents; and for a few topics, they spent less than a minute locating and selecting the clusters of relevant documents. It was extremely easy to gather up the clusters based on the labels of probable content. In most cases, it freed the users from needing to read individual documents at all.

The use of an information mapping tool, in combination with a tool that measures document similarity (which need not be Acquaintance, but can be any system that characterizes the degree of similarity between two documents), is a very powerful method of exploring a database of documents. With such a system, one can understand the overall relationships among the set of documents. Unexpected relationships can be uncovered, and the centrality of certain documents is shown by the way that those documents draw together many disparate document clusters. The usefulness of such tools for dramatically enhancing both text retrieval and knowledge acquisition from a database is just beginning to be realized.

In terms of average precision, Acquaintance scored above the median in ten of twenty-five topics. That is not very impressive. However, informal results of the first task presented at the interactive panel session during the conference indicated that the performance of the Parentage and Acquaintance interactive system was very good, when other factors, such as the time to recover relevant documents, were taken into account.

Filtering

The object of the filtering task was to adjust a text retrieval system in such a way that it retrieved documents with high precision on one run, with high recall on another, and with a balance of precision and recall on a third. The data and queries used for these runs were the same as those used on the routing task.

The Acquaintance system attempted to achieve these three levels of performance by varying both the n-gram length and the threshold at which scores were reported. As n-gram width increases, the system obviously requires longer strings of text to be identical for them to be hashed to the same address in the document vector. By increasing n-gram length, and requiring a higher score threshold for defining documents as similar, the precision of the output should be increased.

For the high recall run, the n-gram length was set to five, and the score threshold was set at 0.25. This is actually close to typical parameters for using Acquaintance for topic based document retrieval. For the high precision run, the n-gram length was increased to seven, and the score threshold was increased to 0.40. This forced more and longer stretches of text to precisely match between the reference documents and the documents from the database to pass the threshold. For the balanced run, the n-gram length was kept at seven, but the score threshold was lowered to 0.30. This actually would result in a somewhat more stringent test for similarity than is normally used, but is still significantly less than the high precision run.

The results on this track were very poor, when compared to the other three systems that participated in the track. This is a reflection of the overall difficulty Acquaintance had with the mismatch in content and style between the reference documents for the routing task, and the documents in the routing database. It is not clear that better performance in comparison to the other systems could have been achieved by adjusting the parameters of the system given that fact.

Confusion track

This was a new track at TREC-4. Instigated in part because of interest by the defense community, this track was created to provide a vehicle for testing how text retrieval systems perform in the presence of garbled data. In the defense and intelligence worlds, data is often received in garbled form. Sometimes the garbling can be quite severe, and a system that cannot deal gracefully with degraded data is very limited in its usefulness.

The data for the corruption track consisted of the category B data, that is, a subset of the TREC data taken from Wall Street Journal and San Jose Mercury News articles. The data came in three forms, ungarbled, randomly garbled at ten percent, and randomly garbled at twenty percent. Random garbling meant that for any character in the text, there was a ten or twenty percent chance for that character to be changed, lost, or an additional random character inserted, with all garbles guaranteed to result in ASCII characters. Samples of the ten and twenty percent garbled text are shown in Figures 3 and 4. Only four systems participated in this track, and only Acquaintance and one other even attempted to process the data corrupted at the twenty percent level.

```
To muc excitSement on top of too much cold iedZcation kmay have caused the
pacpidPfjartbeatt t9haI forced Mansas Cidty linebacker DerricLC1 Thomas out of the
Chiefs'plasyoff game SKturday.;   Thomas, a Pero OBowlhselection in all thre of
hi6s yes in the NF, went out in the second qcartaer of the Chiefs' 10-6
JiTctory over the LoYs Aygeuts RMidYrsshortly Zftey a sakH tht noced a fumQle.
  Sporwzs
 3 RAPID EARTBEAT FORCES THOMAS TOEYAVE GAME
OC. STAR IS EXPECTJD TO PLAY NEXoT WEEKEND
  Pro Football; AF8C Notebook
```

```
        He kas takenito a hospital as a pre1Fution, although his heart rawte was
back to normal by thehtime he leftF the sadium. H7 remained ov9rnight for
observation.;   "The doctors indDMcaed Derrick may have taken too much cold
med7cation beforO te agakzmeJ," Chefs Prsident6Generna MVnager2Crl Peteron
sad. "That combined fwith t4he excitement of thegamef may haveS caused the
prblem.;   "We don'ti think it's anthin to be alarmezdabout.m;   Thomas is
expectdio be yable to play2next weekend.; SECOND-GESSING: RaiXer0s Coach Art
Shell refused to be secGnd-Ogussed about stBrting Todd Marinovichg at
quarterbsck ovr veteranpJay SchDoede.;   "ouF cwm doJit if you want," he
said, "but I'mhnot going to scond-gues mysJ1Uf.B;   SellF also bristled when
aUkd itf he considereT eplcing arinovich with Schroeder lateUin the game.;
"Mysthinking in the fourth 3quarter wisA fthaj we were here with the kid andwe
are going to finishGwi6th him," Shell sai.;   Schreder and YShell Ps5id that
Sch3oede5r, who sprained both ankles two9 weeks ago, was h4ealthy enoughCto
play.; COyTROVBRSIAL LAY: The second of Marinovich'svC fou interceptionsset
up the ony touchdwn oC the game.;   The score came on an 11-yard recepton
by the CXhiefs6' Fred Rones witpi 5 minutes, 7 slconds left i1 the second quartRr
```

Figure 3

Article 1 (SJMN91-06364024) from San Jose Mercury News at 10 percent garbling

```
Too much excitement n tp od toko mucx c3ld medicticon a ave causted9 te
rapid hertbeat thdat forced Kansas Cit  11inRblker Derric Thomaas sout4of te
hiefL' playsvf camee SaOtujaeI;  gThomas, a Prao Bo6wl s7elction ino all thrQ oQ
Eis year in ahe NFL, wen ut in th sTelom quater oYf te Cifs' 0-Y
viNctory oer 6he LoswH Andgl7s Raiderssio2tlIatzer64 sack7that fotNed a fumble.
 7 Y cports
 e  RAPID HEMATBAT FORCES THOMoAS TO LEAVh GAME
K.N. STAR 96S EXPkCTED TO PLAYNEXTWEEKND
   Pro FootballX AF5Notebhok R
    c  jHe was ta9en to anhospital asAa pncaution, alth ough hims heart9b rMte was
beckqto nolmal y3the tie fe left thCsMtaNdium. Hesemaieed ovCrniht for
obsarvrtion.t  1TheV adokzctJors igndibyated rrickn vmy haypotaken tSoo7muce cYld
medicatfion be4fose the gafe," ChWibHfs PrxsSident-reneraklZManager Carl Pieterson
said. "FThaZt combined wzi1th4te excitement of the r5gabe8may havescauNd the
jroblem;   "WLk don'tMthnk 0it's anything ts be4alarmed wbotF"O; Q  Thfjas i
expectedto be ale tog play next eckenud.; SEcND-GUESSNB: IRaidevrFs och At
Shell refusHd tocme seconyussed abouB oiarting TkoddMaUriovich 3at
Fquarterback over vetra4n8Jay Schroeoer.  o "BYou 3can dPiL if you want,"mh
saidi "but I'm nt going tC second-YuUs3s myself.; A f S5ell also britled when
ased if 3he consfdered replaing6 Marinovichwith 7chrder xlate inYthevPaw;
 "My ttnki5ngIin the fourth quarter was t9hat we were here8ith theGkid nd e
are goig D finibfhDwith hiY" Shell said.;Z vC Schkrceder aMd yShell said Jthjt
ScyhrodZr who spraibed both aneles two weeky ago, wxas wealtXhy e9ugJ to
lay.; CO5T ROVERz0AL PLAY The sepd oJMarinovioh'sLfur iterceptions set
0J tdhe only touchdown f the ga6me.;   The scoKeN4 cIme on an 11-yard receWtion
y the Chefs' Fced Jons with 5 m3inUutes, 7 seFconds luft inthe se od quarter
```

Figure 4

Article 1 (SJMN91-06364024) from San Jose Mercury News at 20 percent garbling

The uncorrupted and the ten percent corrupted text were processed in the same manner as the data in the basic ad hoc task. The n-gram length was five for both runs. The only change when processing the twenty percent garbled data was to change the n-gram length to four. This increased the chance that any particular n-gram would remain ungarbled.

Since Acquaintance is statistically based, some "noise" in the data should not cause the algorithm to fail catastrophically. In fact, Acquaintance performed very well on this task. It suffered minimal degradation in recall and precision between the uncorrupted and ten percent corrupted data; at ten percent garbling, Acquaintance scored above the median on thirty out of 49 topics. And while performance dropped again at the twenty percent corruption level, overall, the system still performed quite well. This indicated that the statistical nature of the algorithm let it degrade gracefully, and relatively slowly, as the data became more corrupt.

Spanish

The Spanish track was essentially the same as the English ad hoc track. Participants were given access to the Spanish database early, and then the queries were sent out shortly before the results were due back. The queries were, like the English queries, quite short. The database contained articles from El Norte, a Mexican newspaper.

The problem for Acquaintance here was the same as that when doing the ad hoc task in English. The topic descriptions were so short that they did not provide enough of a statistical profile to properly model the topics. A typical topic (number 32) read "Cual es la importancia de las Naciones Unidas (NU) para Mexico?" To overcome this, the topic descriptions were again manually expanded just from general subject knowledge of the users. Unfortunately, the users do not speak Spanish, and were not knowledgeable about Mexican affairs. Therefore, the "expansions" were very minimal, in fact usually consisting of removing clearly uninformative verbiage from the query rather than adding anything substantive to it. The rendering of the above query became "importancia de las Naciones Unidas (NU) para Mexico." Obviously, "query expansion" was of minimal use to Acquaintance in this track.

The Spanish ad hoc queries were processed in exactly the same manner as the English ad hoc queries. The results reflected the problems with the minimal queries; the performance of the system was quite poor. With fuller topic descriptions, or better manual query expansion, performance would have most likely have improved significantly, and in fact, should have been very comparable to the performance on the English ad hoc task.

Summary

The Acquaintance technique was developed to find documents that are similar to one another, or to a reference document, in a language independent and potentially garbled environment. For this to work acceptably as a topic spotting technique, it needs a modest amount of text in both the reference and the target documents that is relevant to that topic. In TREC-4, the queries in the ad-hoc based tasks were significantly sparser than in TREC-3, and this sparsity of text had an impact on the performance of the algorithm. Even so, the minimal manual augmentation of the topic descriptions, and the strategy of using the most highly connected documents from the first pass as reference documents helped improve the actual performance of the technique to the point that it outperformed last year's ad hoc results.

In the routing task, the documents against which the queries were compared were often either quite sparse and very different in style from the reference documents (the newsgroups), or quite diffuse (the federal register documents). This led to Acquaintance building very poor models from the reference documents of what was in the documents in the databases. The results in the routing-based tracks reflected this mismatch by the very poor performance of the system.

The system did perform quite well in the confusion track, which measures performance in an area where Acquaintance has a high degree of potential, namely, working with garbled data. Even at a relatively high degree of garbling, the system's performance degraded quite gracefully. This type of behavior is quite important to users of document retrieval and filtering systems in the defense and intelligence fields.

The other area where performance was rather good was in interactive document retrieval. This was achieved by the combination of Acquaintance with Parentage. The usefulness of information visualization for text retrieval, when combined with virtually any document retrieval engine, clearly has great potential.

References

[Cohen 1995] Jonathon Cohen: "Drawing Graphs to Convey Proximity: an Incremental Arrangement Method," submitted to ACM Transactions on Computer-Human Interaction.

[Damashek 1995] Marc Damashek: "Gauging Similarity via N-Grams: Language-Independent Categorization of Text," Science **246**, 843-848 (1995).

[Huffman 1995] Stephen Huffman, in preparation.

TREC-4 Experiments at Dublin City University: Thresholding Posting Lists, Query Expansion with WordNet and POS Tagging of Spanish

Alan F. Smeaton[1], Fergus Kelledy and Ruairi O'Donnell

School of Computer Applications, Dublin City University, Glasnevin, Dublin 9,
IRELAND (asmeaton@CompApp.DCU.ie)

Abstract

In this paper we describe work done as part of the TREC-4 benchmarking exercise by a team from Dublin City University. In TREC-4 we had 3 activities as follows:

- In work on improving the efficiency of standard SMART-like query processing we have applied various thresholding processes to the postings list of an inverted file and we have limited the number of document score accumulators available during query processing. The first run we submitted for evaluation in TREC-4 (DCU951) used our best set of thresholding and accumulator set parameters.
- The second run we submitted is based upon a query expansion using terms from WordNet. Essentially, for each original query term we determine its level of specificity or abstraction; for broad terms we add more specific terms, for specific original terms we add broader ones; for ones in-between we add both broader and narrower terms. When the query is expanded we then delete all the original query terms in order to add to the judged pool, documents that our expansion would find that would not have been found by other retrieval. This is run DCU952.
- The third run we submitted was for Spanish data. We ran the entire document corpus through a POS tagger and indexed documents (and queries) by a combination of base form of non-stopwords plus their POS class. Retrieval is performed using SMART with extra weights for query and document terms depending on their POS class.

The performance figures we obtained in terms of precision and recall are given at the end of the paper.

[1] The work reported here was carried out while the first author was visiting the CLIPS-IMAG laboratory in Grenoble during 1995 funded by the Ministère de l'Enseignement Supérieur et de la Recherche and by Dublin City University. Thanks are due to François Paradis (Grenoble) for help with Perl scripts. Thanks are also due to Jim Cowie and staff at CRL in NMSU for use of their SPOST POS tagger for Spanish, at their site and for their help with tagging the Spanish corpus

1. Introduction.

The work reported in this overview is a description of the research we carried out for our efforts in TREC-4. Essentially, we have submitted 3 official runs, the first two for Category A ad hoc, and the third for Spanish ad hoc. Each of the 3 runs are based on 3 separate streams of research and are described independently in the following 3 sections of the report.

For all our work, the platform we used for our experiments was a SUN SparcStation 20 with 96 Mbytes RAM and 12.5 Gbytes local disk attached directly to the SCSI port and a further supply of disk space available via NFS and an ETHERNET LAN.

2. Run Based on Thresholding the Postings Lists

2.1 Thresholding

Under normal processing conditions, the query space in an inverted file is made up a concatenation of the posting lists of query terms. As a retrieval algorithm processes entries in the posting lists, a set of accumulators are used to keep scores for individual documents and at the end of the processing the top-scoring documents, identified by the highest accumulator values, are retrieved.

In our approach to query processing, the query space is composed of query term posting lists sorted in order of increasing posting list length with each posting list also sorted in order of decreasing value to its respective index term [Kelledy and Smeaton, 1995].

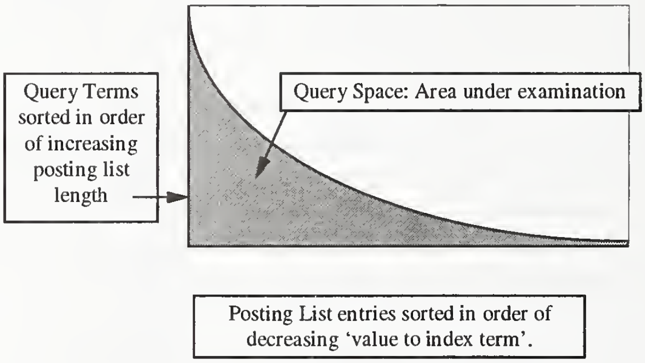

Query Terms sorted in order of increasing posting list length
Query Space: Area under examination
Posting List entries sorted in order of decreasing 'value to index term'.

Figure 2.1 Abstract view of Query Space

Figure 2.1 gives us an abstract view of the Query Space (*QS*). It is this *QS* which must be processed in the most efficient and effective manner as possible in order to respond to user queries with meaningful answers, in this case a list of documents ranked in order of possible relevance to the user's query. Due to the structure of the *QS* in which the posting lists are arranged vertically and horizontally in order of probable value to the query, one can assume that the concentration of relevant information with respect to the query is higher in the top left part of Figure 2.1 than in the bottom right part. This *QS* structure facilitates the notion of thresholding, in which not all of the *QS* need be processed. This results in four main options available for investigation, namely no thresholding, query term thresholding, where only a subset of the terms within the query are processed, posting list thresholding, where not all of the posting information in each posting list is processed and finally, the combination of query term thresholding and posting list thresholding, where only a portion of the posting list information for a subset of query terms are processed.

2.2 No Thresholding

Figure 2.2 illustrates the basic approach to processing the query space. In this variation no thresholding whatsoever is applied. The result of this approach is that all posting entries of all query term posting lists are processed. In a situation such as ours where we could have large queries (on average ~50 terms per query) and a very large document collection the cost of processing the query space associated with a given query is a non-trivial one. This coupled with the fact that a lot of noise[2] would also be generated from processing non-discriminating terms with low within document frequencies (the lower right hand side of Fig. 2.2) leads to relatively poor results. This lower right hand size of the query space is more likely to contain postings which, firstly, are associated with terms that occur in a large proportion of the documents within the collection therefore reducing there discriminating power and secondly, their within document frequency is likely to be very low (due to the sorting process). These two facts make it more likely that postings in this region of the query space for non-discriminating terms with low within document frequency are more prone to generating erroneous information i.e. noise and therefore slowing down overall system performance.

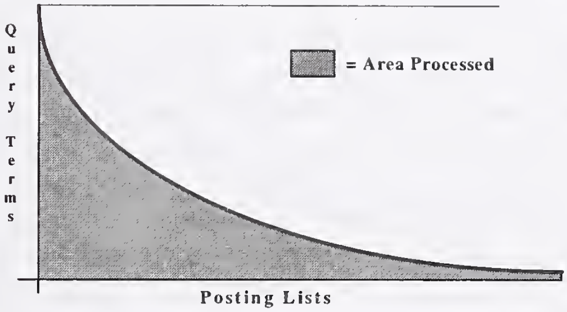


Figure 2.2 No Thresholding on the Query Space.

Figure 2.2 illustrates the overall procedure involved in processing the *QS* without any thresholding techniques in operation. The enture shaded area in Fig. 2.2 is processed and every accumulator that acquires a weight greater than 0 is passed on to the document scoring and ranking procedures.

2.3 Query Term Thresholding

Improvements to efficiency and effectiveness of the retrieval process can be achieved by incorporating Query Term Thresholding (*QTT*) into the processing of the query space when the queries are long. This involves computing the percentage of documents within the collection in which each of the query terms occurs. There is very little additional overhead involved in computing this and these percentages are used to exclude the processing of the most non-discriminating query terms (those that occur in a large percentage of documents within the collection). At the present stage in our experiments the thresholding percentage is fixed for all queries, we hope to modify this procedure to allow the threshold value to be automatically computed based on criteria associated with the query. Figure 2.3 illustrates the net effect of *QTT* on the query space, query terms up to a cut-off point are processed, after which the longer, more

[2] Noise: Erroneous data which might interfere and degrade overall system performance.

time consuming and 'noisy' posting lists are discarded therefore reducing the time taken to process the query space.

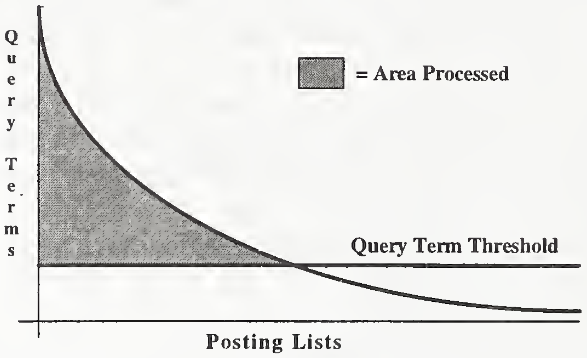


Figure 2.3 Query Term Thresholding the Query Space

2.4 Posting List Thresholding

Further improvements to efficiency can be achieved by incorporating Posting List Thresholding (*PLT*) into processing of the query space. This involves computing percentage of the posting list for each query term that is to be processed either in advance as static values or dynamically depending, for example, on the ranked position of the term within the query. There is little overhead involved in computing these values which in turn are used to exclude the processing of the most non-discriminating posting list entries (those that occur a small number of times within a given document). Figure 2.4 illustrates the effect of *PLT* on the query space where query term posting lists up to a cut-off point are processed, after which the least useful postings for that term are not considered to be of great importance to the query relative to the postings already processed. This reduces the time taken to process the query space by eliminating the need to process all postings entries for a given posting list, it will also reduce the number of document accumulators activated by noisy posting list entries i.e. a document accumulator being activated by a once off occurrence of a term within a document which could possibly have a negative affect of the overall systems performance. This document accumulator once activated must be included in the sorting procedure therefore also having a negative impact of the system's efficiency.

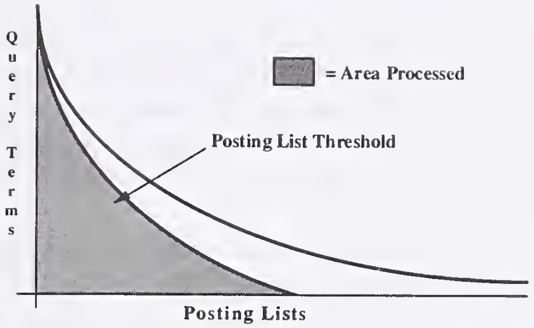


Figure 2.4 Posting List Thresholding the Query Space

2.5 Query Term and Posting List Thresholding

The above two approaches, namely *QTT* and *PLT,* can be combined in order to further reduce the amount of the query space to be processed. The result of combining these two thresholding approaches can be seen in Fig. 2.5. Depending on the parameter values chosen for *nqt'* and pqt_i quite a large reduction in the area of the query space to be processed can be achieved.

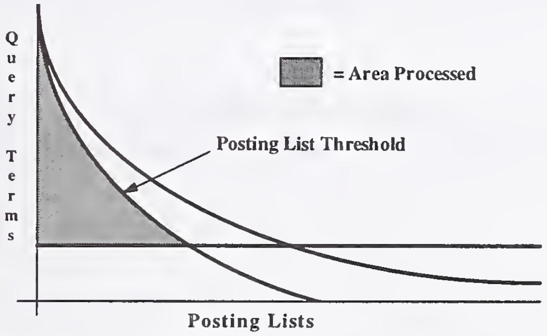


Figure 2.5 Posting List and Query Term Thresholding the Query Space.

It is obvious that the above approaches either taken alone or combined will have a positive effect on the efficiency of a retrieval process. However it remains to be seen what impact the above approaches will have on the effectiveness and the above approaches can only be justified if they do not noticeably degrade overall system effectiveness.

2.6 Restricting Document Accumulators

Processing the *QS* is only one aspect of the retrieval process. The output from the processing of the *QS* is passed to the Accumulator Space (*AS*) which can be visualised as a set of counters, one for each retrieval unit. The unit of retrieval can be a document, page, paragraph, sentence or a word. The more fine-grained the retrieval unit the larger the *AS* becomes. A given query will only activate a subset of the *AS*, by activate we mean store a non-zero value in an accumulator. This Active Accumulator Space (*AAS*) must then be processed. This processing might entail a conversion from one unit of recovery to another, for example, page to document resolution. This means that the individual query page similarity scores are combined in some fashion to form query-document similarity scores. These query unit similarity scores, be they page or document similarities, must then be ranked in order of estimated relevance to the query and presented to the user.

The extraction of the *AAS* from the *AS* and its subsequent processing (unit resolution and sorting) contributes much of the overhead involved is query processing. This coupled with large *QSs* and no thresholding optimisations results in poor system performance in terms of efficiency. In such a situation even if the system is providing high quality results the user may not be prepared to wait the necessary amount of time to receive the response from the IR system.

It is therefore of great importance that attention be given not only reducing the amount of the *QS* that is processed but also the storage structures used to manipulate the *AS*. A simple assumption that can be made at this point is that the more information that ends up in the *AAS* the longer it will take to process the query. Following on from this assumption it is logical to try and control the amount of information that goes into the *AAS*. This control of the *AAS* information can be implemented during the processing of the *QS* by introducing the notion of restricting the number of accumulators that can become active with respect to a given query.

2.7 Experimental Results

The variations on the basic query processing and retrieval algorithm outlined so far require a number of parameter settings in order to operate efficiently and effectively. We turned TREC-3 topic descriptions, generally accepted as being long, into a shortened form and used the shortened form as queries in order t determine the best set of parameters used in *QTT*, *PLT* and the reduced accumulator set.

Figuures 2.6, 2.7 and 2.8 show how setting the maximum number of document accumulators effects the average precision, the interpolated precision at 0.0 and the precision at recall level 0.10 respectively.

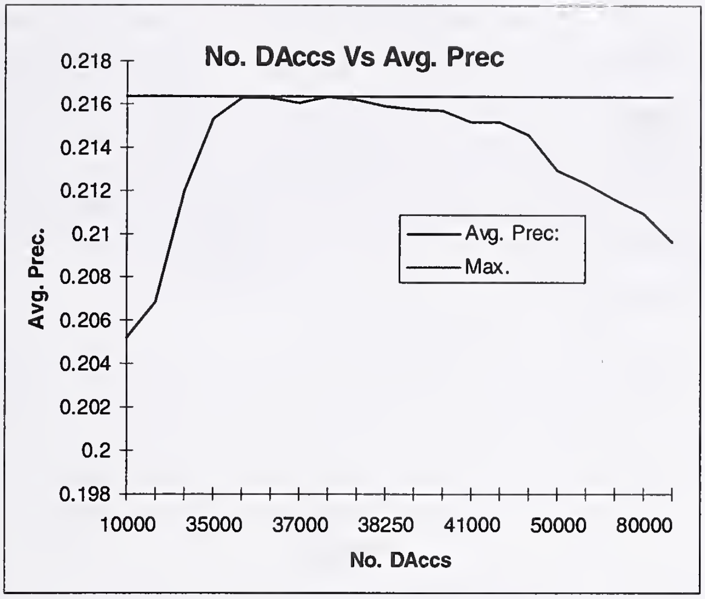


Figure 2.6 No. of Document Accumulators Vs Average Precision

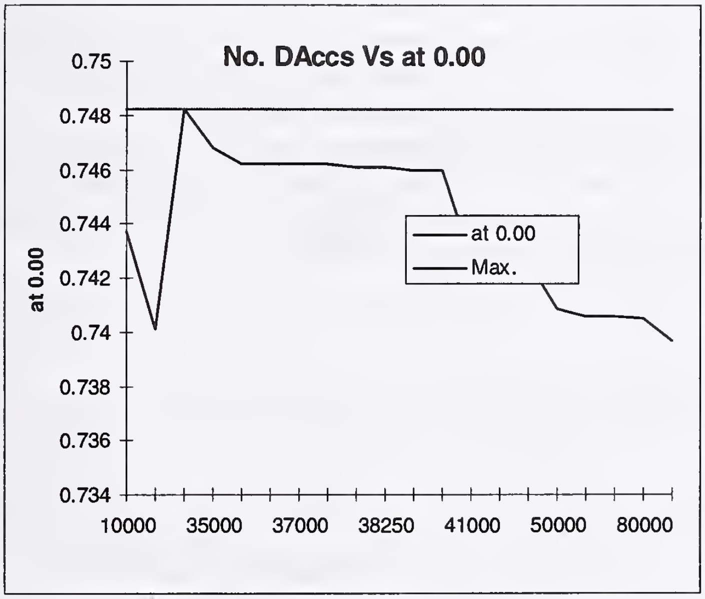


Figure 2.7 No. of Document Accumulators Vs Precision at 0.00

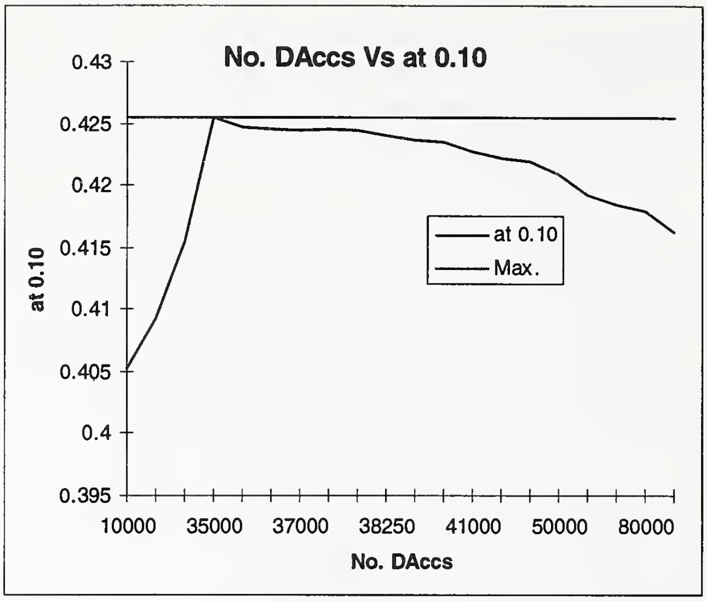


Figure 2.8 No. of Document Accumulators Vs Precision at 0.10

These three evaluation aspects show the peak performance in terms of effctiveness is where the maximum number of accumulators is set to between 35,000 and 36,000 registers. Given that there are a much larger set of documents than this in the collection this is making a considerable impact on efficiency. The big surprise is the effect these restrictions have on effectiveness. It is a positive **improvement** in retrieval effectiveness to introduce such efficiency considerations. This is shown even more dramatically in the graph in Fig. 2.9 illustrating retrieval effectiveness when no thresholding at all is used vs. our best parameter setting. These experiments were run on TREC-3 topics as training for the parameter settings for our TREC-4 runs.

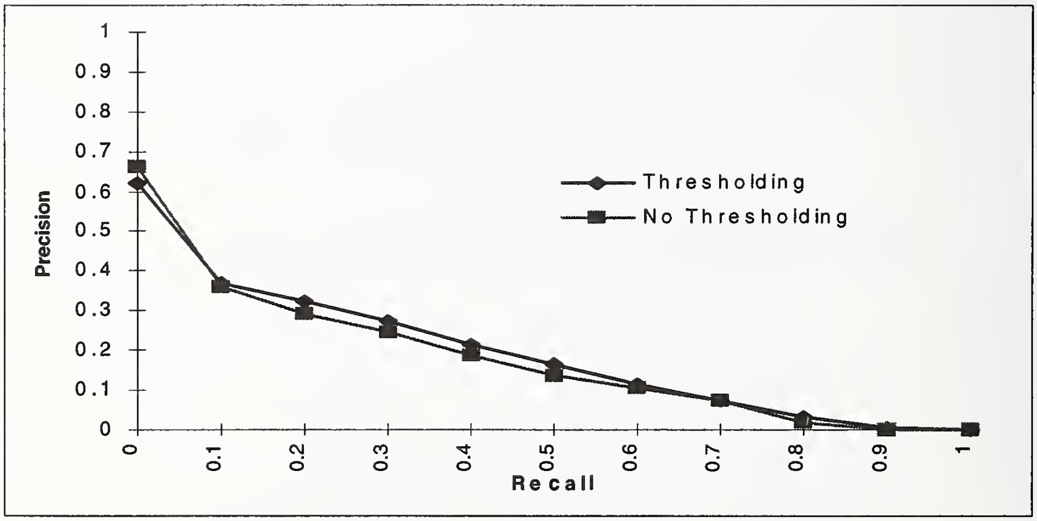


Figure 2.9 Recall-Precision for Thresholding Approaches. (TREC-3)

Here we can see that the thresholding is causing the overall effectiveness to be improved, as well as making improvements in retrieval efficiency. Given these results we settled on the best number of accumulatrors, *QTT* and *PLT* thresholding as determined by training on TREC-3 topics and used these in our TREC-4 run. This run was coded DCU951 and the precision-recall figures are shown in Fig.2.10. Post-official-submission, we re-ran these queries with no thresholding or accumulator restrictions and the results we obtained in that subsequent run is also shown in Fig. 2.10

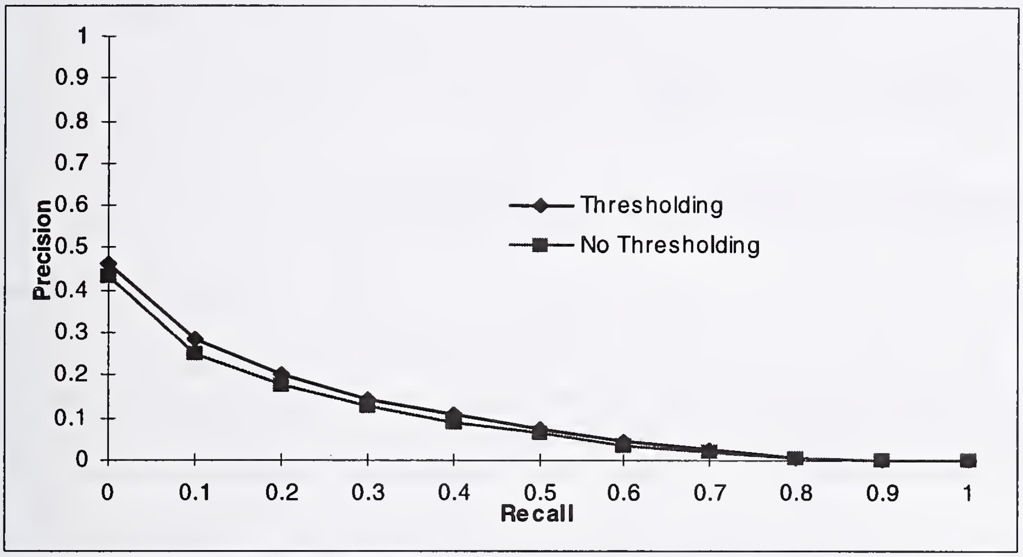


Figure 2.10 Recall Precision for Thresholding Approaches (TREC-4)

	No Thresholding	Thresholding	% Improvement
Postings Processed	58,284,390	29,678,636	49.08%
Elapsed Time (Sec)	5732	853	85.12%
Accumulators Used	25,537,453	1,499,865	94.13%

Figure 2.11 Efficiency improvements (TREC-3)

	No Thresholding	Thresholding	% Improvement
Postings Processed	9,606,861	7,796,370	18.85%
Elapsed Time (Sec)	1,353	452	66.59%
Accumulators Used	7,057,794	1,440,602	79.59%

Figure 2.12 Efficiency improvements (TREC-4)

Figure 2.11 and 2.12 illustrate the improvements in efficiency gained by employing the thresholding techniques outlined above. It can be clearly seen that these thresholding approaches perform better in terms of improving efficiency when operating on larger queries as is the case in TREC-3. While the results achieved using our thresholding approaches on the TREC-4 data are not as impressive as TREC-3 results they still represent a significant saving in the overall time taken to process the queries. The reason for the poorer results in TREC-4 is due to the much smaller queries which leave the thresholding techniques much less room in which to operate. We believe these thresholding approaches will offer maximum benefit when applied to large or automatically expanded user queries.

The overall performance of our system in terms of effectiveness with and without thresholding was fairly poor compared to other. This we attribute to an over simplistic tf*idf weighting scheme coupled with the restriction of only indexing documents and queries by WordNet nouns and verbs (necessary in order to link with our other submitted run). We intend carrying our further work in this area by removing the WordNet noun / verb restriction and replacing the existing weighting scheme with other well known, high performing weighting schemes (OKAPI,Cornell) and determining whether or not our thresholding approaches attain similar improvements in efficiency without adversely affecting effectiveness.

3. Run based on WordNet Query Expansion.

The approach we describe here is based on expanding an original query by adding terms from a structure derived from a lexical database, WordNet. We perform a thesaurus-like expansion of original terms and in doing this we hope to collect relevant documents not found by the more traditional keyword IR approaches. For the purposes of fine-tuning, we trained the system using queries from TREC-3 which are considerably longer than the queries for TREC-4. To account for this we produced shortened versions of TREC-3 queries and used these for experimentation.

Although the query expansion work reported here was carried out in Grenoble, France, the experiments were run on computers in Dublin. Documents have been partitioned into fixed-size pages based on word counts and the "score" for each document is the "score" for the highest-scored partition it contains. This addressed the issue of normalisation for document length.

Each document has had stopwords removed and has been indexed by WordNet nouns and verbs, both single word and collocational, that occur in the text. Adjectives which can only appear as adjectives (*big, long,* etc) are discarded as are adverbs where these do not have verb or nominal forms. Terms indexing documents (or document partitions) are weighted by their IDF weights and a document (partition) score is the sum of the tf*IDF weights of query terms occurring in that document (or partition). The system used to implement this retrieval was written in C by Fergus Kelledy and uses an inverted file for efficiency. It was developed to facilitate research into improving search time by sorting and applying thresholds to postings lists in inverted files as in our first official TREC-4 run. A version of the system which searches the TREC collection and runs in batch mode was developed and the input to runs of this system was a query file which contain, for each query, the query terms to be used. This meant that when we wanted to evaluate a query expansion strategy we generate a new query file containing the query terms we wanted and used this as an input to a run. The query files were developed in Grenoble and sent to Dublin by ftp and then used as input to runs on the machines in Dublin.

The format of the query file was such that it allowed us to assign weights to query terms depending on the query expansion strategy being evaluated. This was done by specifying, in the query file, each term to be used plus a weight for that term to be used as an envelope factor in computing document scores. An individual document's score would be the sum of the tf*IDF weights of original query terms plus 0.5 times the sum of the tf*IDF weights of synonyms. A good deal of experimentation took place varying the contribution of these envelope weights. The time taken to perform a run of the system depended on the number of terms added to queries; the more query terms, the longer the processing. Because of the way in which the implementation worked, the increase in time taken was almost linear. For a set of 50 queries searching c.2 Gbytes of document text with an average of 8 terms per query, a run took between 10 and 15 minutes of elapsed time. Most of this was spent on disk I/O.

3.1 WordNet and HCGs

WordNet is the product of a research project at Princeton University which has attempted to model the lexical knowledge of a native speaker of English [Miller, 1990]. The system has the power of both an on-line thesaurus and an on-line dictionary, and much more. Information in WordNet is organised around logical groupings called synsets. Each synset consists of a list of synonymous word forms and semantic pointers that describe relationships between the current synset and other synsets. A word form can be a single word or two or more words connected by underscores, (referred to as collocations).

In our work on defining relationships between words we only use the nouns and IS-A links from WordNet, ignoring the verbs, adjectives and adverbs. Where a TREC query contains words in other classes these will be nominalised as discussed earlier. For non-noun word occurrences in documents, these will be ignored. This leaves us with c.60,000 word forms of which more than half are single words and the remainder collocations. This is adequate for our task.

The initial form of our KB consisted of a number hierarchical concept graphs, (HCGs), constructed directly from WordNet. The root concepts of the HCGs were chosen as result of a set of experiments to determine what root concepts would, as a group, provide maximum coverage of the nouns in WordNet whilst minimising the degree of overlap between HCGs. The motivation for creating HCGs was purely efficiency in order to make the computation more manageable and as we will see later we have not sacrificed effectiveness in any way. The resulting HCGs ranged in size from 43950 unique concepts (Entity) to 688 concepts (Shape). While we would have preferred a more even balance across HCG size, we believe the disparity in the HCGs we use does not cause any major problem.

The information based approach to measure semantic similarity which we used in these experiments is motivated from work carried out by Resnick [Resnick, 1993] and was done by Ray Richardson [Richardson & Smeaton, 1995]. The semantic similarity of concepts is estimated using the distributional behaviour of classes of words in a large text corpus. Resnick viewed noun synsets as a class of words where the class is made up of all words in a synset as well as words in all directly or indirectly subordinate synsets. Conceptual similarity is considered in terms of class similarity. The similarity then between two classes is approximated by the information content of the first class in the noun hierarchy that subsumes both classes. The information content of a class is approximated by estimating the probability of occurrence of the class in a large text corpus. As such, the similarity of two classes of words, $c1$ and $c2$, can be expressed as:

$$Sim(c_1, c_2) = \max_{C_i}[\log \frac{1}{P(C_i)}]$$

where $\{C_i\}$ is the set of classes dominating or superordinate to both c_1 and c_2, $P(C_i)$ is the class probability of class c_i, and $\log {}^1\!/_{P(C_i)}$ is the information content of class C_i. In our work we took 278 Mbytes of text from the *Wall Street Journal,* tagged this with a parts-of-speech tagger to detect occurrences of nouns, or word occurrences which were potentially nouns, and used the frequency of occurrence of WordNet nouns and collocations to compute the information content of synset classes in our HCGs.

The information based semantic similarity estimator measure is not without weaknesses. Perhaps foremost is the fact that this technique ignores information in WordNet that may be useful namely that only the synonym and IS-A relations are used while the other relation types, which are used effectively by the adapted Rada-like conceptual distance approach [Rada, 1989], are unused. A second weakness is apparent in the method we used to calculate the information content of classes whereby many polysemous words will have an exaggerated information content value. If one takes for instance the polysemous word *'bank'*, then the information content for this word will include all occurrences of the string *'bank'* occurring as a noun in the corpus, regardless of meaning. This is due to the fact that we did not, and indeed cannot yet automatically distinguish WordNet word senses from free text with any degree of accuracy above 65% [O'Donnell & Smeaton, 1995]. This net result of being unable to distinguish word senses gives the same exaggerated information content value to a 'commercial bank' as to a 'river bank'. To further complicate things, given the fact that the information content of a class is defined in terms of the information content of its subordinate classes, superclasses of classes containing polysemous words are similarly over-valued. This disregard of ambiguous words is a particular problem given the fact that classes in WordNet refer to particular senses.

3.2 Disambiguating into Multiple Word Senses

In doing the sense disambiguation of TREC-3 and then of TREC-4 queries, we found many instances of simply not being able to distinguish between WordNet's fine senses. The original query term *prison* for example, has (at least) the following WordNet senses which could both be legitimate in the context of query 151 (Coping with overcrowded prisons)

```
{prison}
        a_kind_of {penal_institution}
                a_kind_of {institution}
                        a_kind_of {establishment}
                                ...
{prison, detention, hold, custody}
        a_kind_of {confinement}
                a_kind_of {state}
...
```

We sense-disambiguated queries into any number of reasonable WordNet senses and found in our training runs that this did not degrade performance and as it was a natural thing to do, we did this for our official run. The figures for this were that in the original one-sense-per-term version for

TREC-3 queries there were 310 query terms while disambiguation into any number of senses yielded a total of 448. A similar ration was found for TREC-4 topics. For our official TREC-4 runs we manually sense disambiguated query terms into as many senses as made sense.

3.3 Expanding Terms

In the expanded form of queries, each query term was expanded independently and equally (in terms of terms added) and all synonyms from the expanded generation synsets were added and all of those weighted equally in turn. For the expansions, parents of query terms (P) were given weights, as were grandparents, children (C) and grand-children (GC). We did not do any expansions beyond these generations.

For example, in (this hypothetical) query the original term is "*prison*" which can be expanded to

> { prison 1.0, penal_institution 0.5, institution 0.25, camp 0.5, work_camp 0.25, prison_camp 0.25, prison_farm 0.25, prison_camp 0.25, internment_camp 0.25, prisoner_of_war_camp 0.25, POW_camp 0.25, college 0.5, house_of_correction 0.5, hold 0.5, keep 0.5, jail 0.5, jailhouse 0.5, gaol 0.5, lockup 0.5, lock_up 0.5, cooler 0.5, pokey 0.5 }

... where prison has weight 1.0, penal_institution is a parent with weight 0.5, institution is its grand parent with weight 0.25, etc., and where *P(), GP(), C(), and GC()* refer to parent, grand-parent, etc. The children of prison who have a weight of 0.5 are camp, college, house_of_correction, hold, keep, etc.

3.4 Selectively Expanding Terms Based On Abstraction Level

In the final run we submitted, we examined the information content value (ICV) [Richardson & Smeaton, 1995] of query term nodes in HCGs and used those values as indicators of whether a query term is abstract or specific. We define a broad term as one which would be high up in the hierarchy ... query terms such as

cause	*0.967*
human	*0.973*
individual	*0.937*
person	*0.973*

with their associated ICVs are abstract. Terms such as ...

minority	*6.085*
ineffectiveness	*6.039*
fishing	*6.562*
asbestosis	*6.085*
killing	*6.340*

with their ICVs are more specific. ICVs vary from zero to 7 (in theory, but our lowest ICV was 0.937 and highest was 7.039 (*food_fish*)). Given we can estimate the level of abstraction of a query term from its ICV, we computed the ICVs for all 448 TREC-3 query term word senses and found the mean to be 3.428 (for the terms *legislation* and *school* respectively). Once again a similar distribution of levels of abstraction was found for TREC-4 queries. Roughly, therefore we thus define broad as ICV<3.0 and narrow as ICV>4.0 with intermediate terms having an ICV between 3.0 and 4.0.

For our official TREC-4 run we submitted a run in which we add in children (1.0) and grandchildren (0.5) of broad terms (ICV<3.0) and parents (1.0) and grandparents (0.5) of narrow terms (ICV>4.0) and for the intermediate terms we add in both parents (1.0), grandparents (0.5), children (1.0) and grandchildren (0.5). In TREC-3 queries there were 135 such QTs expanded with that definition of broadness, and 106 QTs terms expanded with that definition of narrowness, with 207 of the 448 query terms expanded by both parents and children

The formal definition of out TREC-4 strategy is based on an idea adopted from that used in the IOTA system [Chiaramella *et al.,* 1987] and is defined as:

$$Q = \left\{ \begin{array}{l} ((syn(P(QT_i")),1.0) \cup (syn(GP(QT_i")),0.25)) \; \ldots \; iff \; ICV(QT_i") > 4.0 \\ \cup \; ((syn(C(QT_i")),1.0) \cup (syn(GC(QT_i")),0.25)) \; \ldots \; iff \; ICV(QT_i") < 3.0 \\ \quad \cup ((syn(P(QT_i")),1.0) \cup (syn(GP(QT_i")),0.25) \\ \quad \cup .(syn(C(QT_i")),1.0) \cup (syn(GC(QT_i")),0.25)) \; \ldots \; otherwise \end{array} \right\}$$

An important factor to note is that our approach to participation for this run in TREC is **NOT** to obtain the best possible results in terms of precision and recall; there are enough other groups taking part in TREC striving to do this. Instead of mixing together all techniques known to yield improve retrieval effectiveness, our interest in this run is in isolating and evaluating the contribution to be made by query expansion using our WordNet-derived structure. To emphasise this we removed all the original query terms from the queries used in the official runs and thus evaluated based on the expanded queries only.

3.5 Results of Query Expansion Run

As would be expected from any retrieval strategy based on *throwing* away the user's original query terms, we performed quite badly in terms of precision and recall figures as shown in Table 1, reproduced from the official results. The interesting thing to note is that for the 49 topics, we retrieved a total of 347 of the 6501 relevant documents, meaning that there are 347 relevant documents who have no query terms, as determined by our indexing into WordNet nouns, at all. It will be interesting to examine the files submitted by other TREC participants to see if we retrieved any relevant documents which were not found by other strategies at all. This is the real evaluation of the strategy we adopted in DCU952. Meantime, we will use the relevant document set determined for TREC-4 topics, to evaluate the query expansion strategy in more realistic experiments based on keeping rather than discarding user's original query terms.

4. Ad hoc Spanish Runs.

TREC-4 was the second time that Dublin City University submitted runs for ad hoc Spanish retrieval.. In TREC-4 we had decided to do something which involved POS tagging of the document and query texts.

About 3 years ago we ran some experiments on the (English) CACM test collection in which we tagged all of the tokens in the documents and queries [Smeaton, 1992]. We then performed a standard SMART-like retrieval based on term weighting but increased the weight for a query and for a document term depending on whether the term occurred as a headnoun, modifying noun, verb, adjective, adverb or stopword. These POS categories are quite rough, for example we did not distinguish between different forms of verbs, but were adequate for our needs. In subsequent experimental runs, which we never published except as an internal report, we were surprised to find that when we doubled the *tf*IDF* weight for adjectives we obtained the best of our results in terms of precision and recall. Our Spanish efforts in TREC-4 were based on repeating those experiments.

Our first task was to locate a POS tagger for Spanish and we had two offers. The first came from UMass who were developing one and the second came from NMSU who had already developed SPOST. As neither tagger had been evaluated in terms of accuracy, after consideration we opted for SPOST on the grounds that it would be different to the UMass tagger and one of the overall objectives in TREC is to build a test collection so with a different tagger to UMass it would be more likely we would retrieve differing document sets that would be better. Problems arose in using SPOST in that it is tied to a Spanish dictionary and in order to move the code to Dublin we would have had to buy a licence for the dictionary which we did not have funds for. In the end we opted to tag the corpus using machines at NMSU, during their night-time period (close to our day given the time difference). We chopped the corpus into manageable sized chunks and tagged each chunk on a different workstation in *nice* mode. At one stage we were running on 8 SUN workstations. The output of the tagger was post-processed at NMSU using code we developed in Dublin, compressed and ftp-ed back to Dublin. There it was used as the input to SMART which had been tailored to allow a document to be indexed by a baseform of a word and the associated grammatical category. Retrieval is based on the usual SMART *tf*IDF* weighting of terms and document ranking.

For retrieval we tagged the Spanish topics in the same was as the documents and input them to SMART as terms and their grammatical categories. As there is no large reliable set of topics and relevance assessments for the Spanish data we submitted runs where the base forms of words which are true adjectives are given double their normal *tf*IDF* weights. When the Spanish relevance assessments are made available we will perform a series of runs varying the categories that get weighted and the amount of those weights and we expect to have an updated result set, in addition to the results of the official run, by the time of TREC-4.

We were initially rather surprised by the poor performance of our approach to ad hoc retrieval in the context of performances from other groups. Our performance figures are given in Table 1 as DCUSP0. What surprised us was the rate at which our precision tailed off. In subsequent, post-official runs we have not been able to move out of this ballpark in terms of effectiveness. This leaves us a long way behind approaches to Spanish TREC ad hoc based on simple stemming and term weighting, an approach taken by the group from Cornell, for example. The following explanations are all possible:

- baseforms derived from POS tagging, while being true baseforms, are not as good as word stems, even using a crude stemming algorithm
- the accuracy of the POS tagger has not been evaluated
- we discovered, after submission of results,[3] that verb and nominal forms of words may not have been reduced to the same baseform in the way we used SPOST. For example, in English, one of the the verb forms of *"to run"* would be *"runs"*, as in *"he runs for the bus every day"* and a nominal form would be *"runs"* as in *"he likes to have his runs every day"*. Using stemming, these occurrences would be reduced to the same term but the way we using the POS tagging meant that these would not have been identified as the same base form (which they are not) or even the same concept (which they are).
- the variant of the term weighting scheme we used, *tf*IDF* weighting from SMART, was not experimented with at all in our experiments

Recall	DCU951	DCU952	DCUSP0
0.00	0.4640	0.0577	0.4066
0.10	0.2907	0.0136	0.2120
0.20	0.2040	0.0061	0.1356
0.30	0.1471	0.0042	0.0967
0.40	0.1093	0.0006	0.0775
0.50	0.0760	0.0001	0.0427
0.60	0.0453	0.0001	0.0202
0.70	0.0246	0.0001	0.0129
0.80	0.0050	0.0001	0.0093
0.90	0.0000	0.0000	0.0000
1.00	0.0000	0.0000	0.0000
Av. Precision over all reldocs	**0.1066**	**0.0037**	**0.0735**

Table 1: Official Results for DCU runs

5 Conclusions

Participation in TREC-4 has been a rewarding experience for the team from Dublin City University. In TREC-3 we participate as category B using the reduced data set and managed to "complete the course", albeit with poor results. In TREC-4 our participation was managed comfortably within deadlines and we succeeded to pursue three differing lines of research. Instead of mixing as many ingredients to improve retrieval effectiveness, however, in each of our differing lines we concentrated on one aspect only.

[3] This pointed was highlighted for us by Ted Dunning.

For the future we will continue to participate in TREC, concentrating on efficiency issues and perhaps also exploring the corrupted data track.

References

[Chiaramella *et al.,* 1987] "A Prototype of an Intelligent System for Information Retrieval; IOTA", Y. Chiaramella *et al., Information Processing and Management,* 23(4), 285-303, 1987.

[Kelledy & Smeaton, 1995] "Thresholding the Postings Lists in Information Retrieval", F. Kelledy and A.F. Smeaton, Proc BCS-IRSG Colloquium, Crewe, April 1995 (also available as http://www.compapp.dcu.ie/CA_Working_Papers/wplist95.html)

[Miller, 1990] "Nouns in WordNet: A Lexical Inheritance System", G.A. Miller, *International Journal of Lexicography,* 3(4), 245-264, 1990.

[Rada *et al.*, 1989] "Development and Application os a Metric on Semantic Nets", R. Rada *et al., IEEE Transactions on Systems, Man and Cybernetics,* 19(1), 17-30, 1989.

[Resnick, 1993] "Selection and Information: A Class-based Approach to Lexical Relationships", P. Resnick, *PhD dissertation, University of Pennsylvania,* 1993.

[Richardson & Smeaton, 1995] "Using WordNet in a Knowlede-Based Approach to Information Retrieval", R. Richardson and A.F. Smeaton, Proc BCS-IRSG Colloquium, Crewe, April 1995 1995 (also available as http://www.compapp.dcu.ie/CA_Working_Papers/wplist95.html)

[Smeaton, 1992] "An Evaluation of Retrieval Performance Using Simple Statistics and SIMPR Linguistic Processing on a Standard Collection of Texts", A.F. Smeaton, *SIMPR Project Report 50.21, Dublin City University,* March 1992.

[Smeaton & O'Donnell, 1995] "A Word Sense Disambiguation Algorithm Using Word Distances; Evaluation on SemCor", *forthcoming, 1995.*

The Excalibur TREC-4 System, Preparations, and Results

Paul E. Nelson, VP of Text Products
(410)-740-8800
paul_nelson@cq.com

Background

This paper describes the system, preparations, and results for the Excalibur text retrieval system used for the TREC conference. After a brief company history and background, we will discuss the system architecture, TREC-4 preparation, an analysis of our results, and some general conclusions.

This will be the third Text REtrieval Conference (TREC) attended by the Excalibur development team. For TREC-1 and TREC-2 we participated as part of ConQuest Software. Please refer to the earlier conference proceedings (available from the National Institute of Standards and Technology) for a discussion of our earlier results in these first two conferences. We did not participate in TREC-3.

ConQuest merged with Excalibur Technologies Corporation (the combined company is Excalibur) in July, 1995, just before the TREC-4 results were due. We are fortunate to have some additional resources to devote to query evaluation, to accuracy studies, and to our preparation and participation in the TREC conferences.

Officially, the ConQuest server system is now part of a larger product suite available from Excalibur called RetrievalWare. We ran TREC-4 with early versions of RetrievalWare 5.0, which is now fully released and available as of December 1995. Many of the improvements and advances which we discovered as part of our TREC-4 evaluations (term grouping, new semantic network weights, a new relevancy ranking formula, and new fine-rank weighting windows) have now been incorporated into Version 5.0 of the product.

ConQuest Software was founded in 1990 with the goal of using Natural Language Processing (NLP) and available linguistic data (such as dictionaries, thesauri, and semantic networks) to produce text retrieval products with high accuracy and performance for large databases and large user populations.

Excalibur was founded in 1980 to create products that utilize Adaptive Pattern Recognition Processing (APRP™) to resolve user queries even in the face of unpredictable and erroneous data (in particular, errors due to the process of Optical Character Recognition when scanning and loading paper documents).

The scope of both companies has grown over the years. The combined company, Excalibur, now has products for document management and high-performance text retrieval, for workgroups, enterprises, and on-line systems. Adaptive Pattern Recognition Processing, in addition to being fully incorporated into RetrievalWare, is also being applied to similar TREC-like tasks on images, such as fingerprint recognition, face recognition, and positive ID.

Finally, we would like to give many fervent and heartfelt thanks to the RetrievalWare development team in the Excalibur Columbia office, who committed long hours and seemingly inexhaustible energies to the TREC-4 task.

Retrieval System Architecture

This section gives a brief description of the RetrievalWare system architecture used for TREC-4. Additional detail can be found in our earlier papers for TREC-1 and TREC-2, or by calling Excalibur Sales at (703)-790-2110.

The basic components of RetrievalWare are shown below:

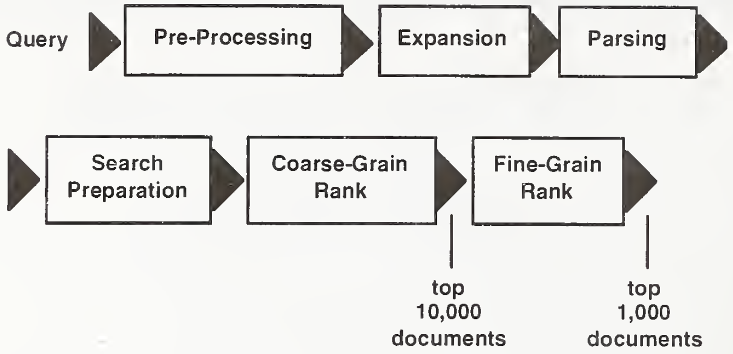


Figure 1 The RetrievalWare Search Engine

Each of the blocks has a well-defined purpose and contribution to search accuracy or performance:

Pre-Processing

Perform query string pre-processing steps to enhance and clarify the characters, tokens, and words in the query. This includes:

Tokenization - Dividing strings into words.

Dictionary Lookup - Locating words in the dictionary and augmenting the words with dictionary information (such as meanings, part-of-speech, inflections, other words with common roots, and variant spellings).

Word Reduction - When a word is not found in the dictionary, rules are applied to reduce the word to a root or simpler form which can then be found in the dictionary.

Token Typing - Identifies the types of special tokens in the query, such as numbers, expansion operators, or other query operators.

Expansion

Apply one of the following expansion techniques to the word. Version 5.0 only applies one of these expansion techniques to any given word (this can be controlled by the user). Future versions will likely apply multiple techniques. Note that the query structures produced by these techniques may not necessarily be the same.

Semantic Expansion - Meanings of the word are expanded to include additional words which are semantically related to the meaning.

Fuzzy Spelling (APRP™) - The word is expanded to include additional words which are similarly spelled.

Wildcards - The user can specify word patterns using wildcards. Words which match the pattern are added to the query, up to a user-defined limit.

Parsing

Query parsing is required to handle a wide variety of special query operations supported by RetrievalWare. These include the following:

Numeric and Date Ranges - Allows the user to search for numeric or date ranges in the body of the document, including open-ended ranges such as greater-than and less-than.

Term Grouping / Query Structure - The user can create statistical groups of terms which are combined together before being combined with other groups or terms. This grouping provides query structure which enhances the "completeness" and "contextual evidence" factors of the query. See the discussion on relevancy ranking below for more details.

Boolean Expressions - Standard boolean queries are supported, including AND, OR, NOT (unary and binary), parenthetical groupings, WITHIN (word proximity in any order), and ADJ (word proximity with order enforced).

Fielded Queries - Any query can be applied to any pre-defined field or zone of a document. Fields or zones of documents are parsed during document indexing with SGML, by using the RetrievalWare document parser, or as defined by a document loading program.

Exact Phrases - Searches for a simple sequence of terms, in the proper order, with all terms adjacent to each other.

User Feedback

At this point the user can view how the query was constructed and expanded and can modify it to better represent the objective. At this time, tools are available to do the following: 1) delete words, 2) choose meanings of words, 3) choose weights for terms, and 4) delete word expansions. The TREC-4 experiments utilized all of these user tools.

Note that this feedback is performed **without** referring to the text of any document retrieved (this is required by the rules for the Adhoc test in TREC). Instead, the query is pre-processed by RetrievalWare and these adjustments are made before the query is executed against the database.

Search Preparation

The search preparation phase is essentially the setup for executing the query. This includes allocating the necessary memory, looking up all words in the indexes, and checking to make sure that all words are found.

Coarse-Grain Rank

Many text retrieval systems use two passes to retrieve documents and RetrievalWare is no different. Our first pass is called Coarse-Grain rank, and it is used to reduce the search set from the entire search database down to a manageable number of documents (typically a few thousand for the highest accuracy, or a few hundred for higher performance). Once the search set is thus reduced, each document can be processed with more accurate statistics with Fine-Grain rank. The two-pass technique is used solely to improve search speed.

The Coarse-Grain ranking algorithm is necessarily more inaccurate than fine-grain ranking, primarily because it considers only the presence or absence of a term in a document (without considering the proximity of words to each other, or the frequency of terms within a document). Coarse-Grain ranking includes the following functions and statistics:

Expansion Term Distance - Documents with terms that are more closely related (by spelling distance or semantic distance) to the original terms in the query are weighted higher.

Query Structure, Completeness, and Contextual Evidence - Once the set of terms in a document is known, the weights of the individual terms are combined using the structure in the query (sets of terms and the functions applied to these sets).

Our algorithms automatically produce structure which enhances **completeness** (documents which contain representatives from all main terms in the original query are weighted higher) and **contextual evidence** (terms are weighted higher when they are supported by expansion terms that are semantically related to meanings chosen by the user).

Main Term Weighting - The term weights chosen by the user, or through automatic statistical techniques, are used to combine term weights together, for the terms which occur in a document.

Fine-Grain Rank

Once a subset of documents has been found by Coarse-Grain rank, Fine-Grain rank uses all additional data to come up with the final set of document statistics. The additional data is primarily the list of positions for every query word in every document. This data is used to compute several statistics, including the strength of each term (based on how far away the term is from other query terms), the average strength of all terms for each document, the maximum term strength for a document, the number of query term occurrences in the document, and a simple count of all the terms in the document.

Finding the strength of each term in the document is (frankly) an area of active research. We experimented with a large number of functions for TREC-4. All functions re-use the same query structure and term weights used for coarse-grain ranking. However, the terms may be combined with different functions, and the term weights may be attenuated based on a proximity weighting window (terms near the edges of the window typically have lower strengths).

Many architectural decisions on RetrievalWare were based on factors which had nothing to do with text search accuracy. These include, royalty requirements and product rights to available dictionary and semantic network information, speed of indexing, requirements for simultaneous indexing and query, and index compression factors.

The most important architectural decision was to make RetrievalWare a full parallel processing text search system that can distribute queries across multiple physical machines and databases in a Local Area Network or Wide Area Network. Distributed queries requires that certain global database statistics be avoided, so that documents retrieved from multiple parallel engines can be easily and properly merged into a single result list for the user. In particular, our system eschews the Inverse Document Frequency statistic (the total number of documents which contain a word, divided by the total number of documents in the database).

Preparation for TREC-4

We strove for two basic goals, which we felt would improve our accuracy the most in preparation for TREC-4: 1) run as many tests as possible to test many variations, and 2) constantly question and refine the results analysis. Thus our approach was more methodical and experimental, rather than theoretical. The raw performance of our engine allowed us to run hundreds of tests and tens of thousands of queries. This allowed us to use some clever numerical techniques to fully optimize search parameters.

All accuracy tests (except for the final results submission, of course) were performed with data from TREC-3 data exclusively. Queries were created using only the "Description" field from TREC-3, to most closely simulate the queries from TREC-4 in size and content. We were worried that we might be hurt by tuning to TREC-3 queries since they are slightly longer than TREC-4 queries (10-15 words instead of 5-10 words), but it appears that this had minimal affect. (See the next major section describing our final test runs for more details on how the final queries were generated.)

All through our preparations, we wanted to double-check our results by testing against TREC-2 queries to make sure that our modifications were not being biased by the TREC-3 query suite. Unfortunately, we were only able to run this double-check once, near the end of our preparations, and the results were (at the time) inconclusive.

The following subsections describe the kinds of experiments that we ran.

Relevancy Ranking

Much experimentation and statistical analysis was directed towards finding the optimal combination of the basic output statistics from the query engine: Maximum Hit, Coarse Rank, Old Fine Rank, and Hit Percentage (Number of Hits / Document Length).

Many techniques were tried for determining "optimal" coefficients, including: 1) a statistical regression over all queries (similar to probabilistic query techniques), 2) regressions with "normalized" statistics (attempting to account for differences between queries), 3) regressions over each query followed by averaging the coefficients, and 4) numerical/iterative techniques to search for the optimal combination. The best technique used the averaged coefficients, with slight adjustments made by iterating the coefficients.

The final function was roughly 4.0*(hit percentage) + 0.59*max_hit + 0.76*coarse_rank. Unfortunately, that regression formula often provides numbers greater than 1. This is because we used linear regressions to compute the coefficients, which attempted to predict a 0 or 1 event (the document relevancy). The regression necessarily overcompensates for the 1.0 case, and so minimum error occurs when the best documents are ranked greater than 1.0.

The five statistics mentioned at the top of this subsection were not the only numbers with which we experimented. We also tried many other functions, including density functions, density overlaps, judging the importance of each term to the document, inverse document frequency, and simply taking the log of the existing statistics. With all these different numbers, there was considerable debate about which ones to use. We tried several techniques for determining which input variables would produce the best results, (such as using the R-squared values to guide our selections, etc.), but no algorithm appeared to be very reliable. Remember that the computation of the coefficients is more complicated than normal because the 50 individual queries were regressed separately and then the coefficients were averaged together, which (we believe) prevented standard techniques from helping much. Ultimately, we chose the statistics above based more on a gut feeling rather than any repeatable technique, but even so we had gained a fair confidence that these were good ones. Obviously, more research is required.

Controlling Expansions

One way of significantly improving our accuracy was to more carefully control which semantically-related expansion terms were added to the query. This appeared to improve our scores by about 4 percent or so (a strong contributor).

Three techniques were used to control the expansions of semantically-related query terms. First, users (query writers) were asked to choose the most appropriate meanings for each significant (i.e. non-stop) word in the query. This limited expansions to only terms which were semantically related to the chosen meanings, based on how the original query terms were used in the query.

The second technique allowed the user to further choose from the list of actual expansion terms for each query term. As it turns out, the expansions for a term contain many subtle shades of meaning, and choosing from these shades of meaning can considerably improve the search accuracy. For example, “murder” expands to “assassinate”, which is probably not useful unless the query has a political connotation. A second is example is that “kill” expands to “suicide”. This technique is especially useful at high expansion levels.

The third technique was to change the weights on the links between the words. Changing the weights not only changes the strengths of the expansion terms, but if the strength is too low (lower than the expansion threshold) then the term will be removed from the query completely. We tried several methods for optimizing the weights. One technique created a TREC-like database of related terms used to evaluate the relevancy of our expansion terms. This database was used to evaluate expansions from all link types, and the weights were set based on the number of relevant terms retrieved. The terms used for this test were based on TREC-2 query terms. This is the technique we used for our formal TREC-4 tests.

Other techniques for evaluating expansion terms included iterating the weight on each link type and then re-running the entire test (which took many days to complete). We also tried simple global modifications such as multiplying all weights by 2, 3, and dividing them. None of these other techniques produced a significant improvement.

Term Weighting

Term weights for our TREC-4 system came from two sources: First, weights for expansion terms were determined by the links traversed in the semantic network (see above). Second, all terms could be manually weighted by the query writer. Careful term weighting appeared to improve the scores significantly, about 5-10 percent.

Initially, we were very skeptical of manually weighting terms, since evaluation of our TREC-2 results showed that manual term weighting did not improve (and may have hurt) the overall system performance. In particular, statistical analysis showed that documents which contained terms that were weighted higher by the query writers had little or no additional probability of being relevant. Needless to say, this result was non-intuitive, and it now appears to have been wrong, since the results from TREC-4 are unmistakable. Most likely, our TREC-2 queries were just poorly weighted, or perhaps our algorithms for evaluating probability of relevance had bugs in it.

Fine-Grain Ranking

Fine grain ranking actually considers the positions of the query terms within the documents retrieved by the coarse-grain ranking function. Terms which are in close proximity are ranked higher by the fine-grain function.

The fine-grain ranking algorithm calls for a window to be passed over the document (only the indexes are required for this step, the document itself is not actually retrieved). For every window position, the query words which occur within the window are combined together (using the same functions as for coarse-grain rank) and the strength of the position is determined.

To further enhance the proximity test, the terms near the center of the window have a higher strength than terms near the edges of the window. A sample window attenuation function is shown below.

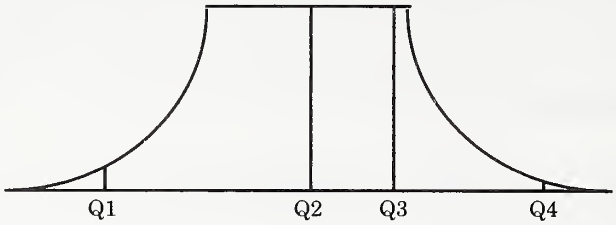


Figure 2 Sample Window Attenuation Function over Four Query Words

In preparing for TREC-4, we experimented with many fine-grain ranking functions, mostly having to do with wider weighting windows and less attenuation of terms which are away from the center of the window (the plateau shown above). Most experiments showed little or no improvement. We ended up using a window which showed a small 2-3% improvement in the results.

The existing system is limited to windows of approximately 200 words wide. Later versions of RetrievalWare may allow us to expand the window to greater than 200 terms (without loss of search accuracy), which may yield further improvements.

Query Structures

Many tests were devoted to improving the structure of queries. In RetrievalWare, queries are not simple lists of terms (as in some Vector approaches), rather queries have structure. RetrievalWare 5.0 was the first version of the software to allow an arbitrary hierarchical structure to the queries (previous versions limited statistical queries to only two levels of hierarchy). Further, RetrievalWare 5.0 allows the integrator to choose the operators (boolean or statistical) which occur at any hierarchical node.

The query structure which helped the most was term grouping, which combines multiple related terms into sets. Each set is ranked individually (as if the set was retrieved from the dictionary). This technique seemed to improve the queries by 5-8%, and also gave us the flexibility to add terms to the query which we might not have added before.

Term grouping is a query structure which essentially enhances the contribution of completeness to the accuracy algorithms. When terms are in groups, the query engine emphasizes documents which have at least one term from each group. Additional terms from the same group do not add as much strength as would additional terms from different groups.

Term groups in RetrievalWare were created manually, and only for the CnQst2 run. Of course, the dictionary naturally creates term groups made up of related terms. This additional grouping is for terms which would not have been retrieved from the dictionary (or for related terms in the query itself).

Things Which Didn't Work

For every test which worked, there were (easily) 30 tests which did not work. More often than not, we would end up with inconclusive data, meaning that the results did not improve or degrade by a significant amount.

Among other things, we were unable to show positive benefits with any of the following:

- Inverse Document Frequency (IDF)
- Expansion normalization
- Other coarse-rank functions (max, min, stronger, weaker, etc.)
- Dozens of other results sorting algorithms
- Miscellaneous hit density functions

The most surprising result here is that IDF showed no reliable improvements to the results. We tried many different ways of weighting terms with IDF, and none seemed to help much. It was suggested at the conference that our other techniques (such as controlling expansion terms, term weighting techniques, and term grouping) may have reduced the need for IDF. It is likely that this is true.

Our Final Test Runs

Excalibur participated in the Adhoc, manually generated, query test over the Category A data. For readers who are unfamiliar with the TREC tests, the “adhoc” test specifies that queries are generated without the benefit of reading documents (i.e. no document feedback), the queries are executed once, and then results are sent to the TREC committee. “Manually generated” simply means that query creation had some human input. “Category A” data is the entire TREC-4 data set, roughly 2.5 GB of text data.

Excalibur submitted two query runs which differed in the amount of manual intervention in query construction. These two query runs are described in the next two subsections.

Query Construction Test 1 (Run ID CnQst1)

The first query construction test (labeled CnQst1) was similar to RetrievalWare expert mode queries: limited to choosing meanings, weights, and expansions. The original TREC-4 queries, with no modifications, were entered into the search system. After pre-processing, the user was given the opportunity to choose the relevant meanings of each word. Then, after expansion, the user was given the opportunity to remove any irrelevant expansion terms from the query. Finally, the user was allowed to change the relative weightings of the terms in the query.

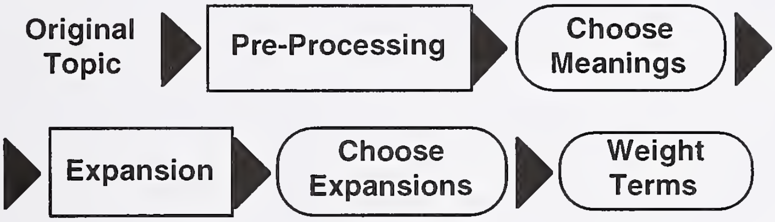


Figure 3 Query Construction for the CnQst1 Run

This method of query construction was chosen because it most closely resembles the standard capabilities provided with RetrievalWare 5.0, as if the user entered exactly the query specified by the TREC committee. Graphical user interfaces are provided for each of these steps.

In our experience, choosing meanings and choosing expansions both turn out to be pretty obvious and straightforward activities. For most words and meanings it is usually clear what is relevant or not.

Term weighting, however, is more complicated and required some experimentation. It was important to identify the terms which were essential to the query and those which were merely supportive. This drove the term weights which were assigned.

Each query took about 5 minutes to create.

Query Construction Test 2 (Run ID CnQst2)

The second run submitted by Excalibur allowed additional flexibility in constructing the queries. Specifically, the query writer was allowed to add terms to the query and was allowed to group terms into phrases or statistical sub-groups (see the discussion of query structures above).

Additional terms were added to the query at the discretion of the query writer. These terms were mostly obvious omissions from the original query, although some terms provided additional examples or supporting information. The query writers did **not** have access to any additional resources when generating these additional terms. In particular, they did **not** use any reference materials or documents. Only terms which occurred to the developers while writing the queries were added.

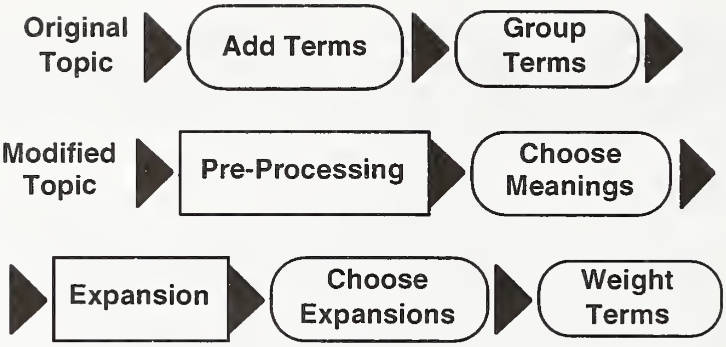


Figure 4 Query Construction for the CnQst2 Run

Term grouping allowed query writers to add more terms than normal to the query, as long as the additional terms were grouped with other terms into main conceptual groups. Basically, a term group behaves the same as if the terms were retrieved from the dictionary as expansions on a concept. The same relevancy ranking formula (essentially a probabilistic combination) is applied to both situations. Since the formula is linear, this means that dictionary expansions of terms within a set are treated simply as if all the expansion terms were part of the set.

Grouping query terms was pretty easy, since the query writers simply created a group for each main concept in the query. Additional query terms were then added to the most appropriate group. Since the TREC-4 queries were so short, identifying the main concepts was quick (the same technique in TREC-2 was much harder, for example, since the topics were so much larger and more complex).

Each query took about 8 minutes to create.

The following are two sample queries used for TREC-4:

```
(recycle:3 recyclable recyclablity retread convert) tires:3 (economic:2
economical:2 economy:2 profitable:2 cost) (Bandag landfill impact)

("bio conversion":5 convert:2 conversion:2 transform:2 generate generation
produce production cogener* "co generation")
```

Conclusions

The following two charts show the performance of both our runs, CnQst1 and CnQst2, in relation to all Manual, Adhoc, Category A systems:

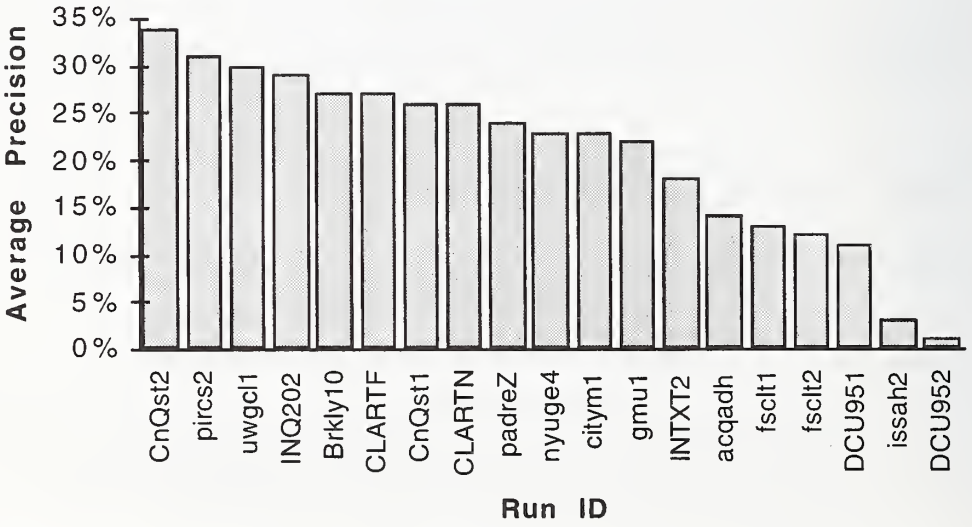


Figure 5 Comparison of Average Precision for all Manual, Adhoc, Category A Systems

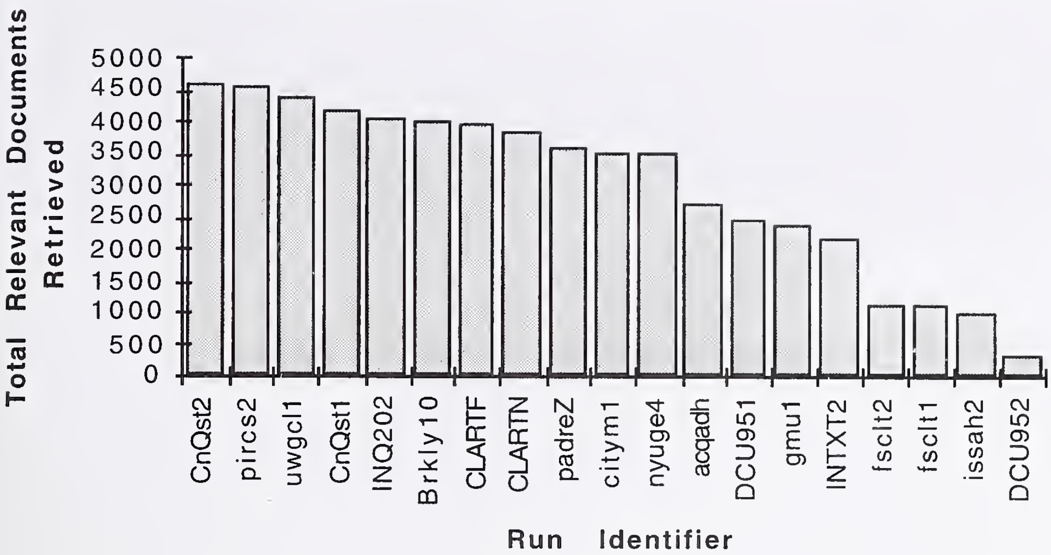


Figure 6 Comparison of Total Relevant Documents Retrieved (Overall Recall) for all Manual, Adhoc, Category A Systems

In TREC-2, Excalibur/ConQuest performed better in Recall (the chart of Total Relevant Documents Retrieved above) than in average precision. Therefore, we were surprised to see that the opposite was true in TREC-4. It appears that our precision has improved considerably, which indicates that our in-depth work on relevancy ranking formulae was worthwhile.

In looking at the two graphs above, it is interesting to note that the CnQst1 run was better in total documents retrieved (recall) than in average precision, in relation to other systems. In fact, it was the only run in the top 9 which changed position.

This leads to another comparison. The difference between CnQst1 and CnQst2 in overall recall appears to be just 9.3%, whereas the total difference between the two runs in average precision was 23.5%. This is pretty surprising. We naturally expected that the CnQst2 queries would be much better in recall, due to the added terms. This implies that additional query terms did not help that much (9% improvement in recall), but that the query structure (i.e. the grouping of terms) helped a lot (24% improvement in average precision). Of course, the difference could simply be due to the scale of the measurements or the sensitivity of average precision (see the next paragraph). This idea will need to be tested further.

During our preparations for TREC-4, we used the total number of relevant documents retrieved as the determining statistic. This is shown above in Figure 6 (there were a maximum of 6501 relevant documents available to be retrieved from the database). We felt that the average precision score was too sensitive to unread documents, especially at the top of the returned results list. This observation came from a comparison of our system variations, where the total relevant documents returned would generally increase while the average precision would vary widely. However, it is unlikely that this instability is responsible for the difference between total recall and average precision in our final runs, since the top 100 documents of each run were read and evaluated by the TREC assessors (and hence the problem of unread documents at the top of the range is minimized).

Two additional comments:

- Only 1 out of every 30 tests that we ran in preparation for TREC-4 actually improved accuracy. Most tests showed little or no performance improvements.
- It appears that the Excalibur/ConQuest relevancy ranking has always performed better on shorter queries. It is certainly the case that our early designs emphasized this fact, primarily because the major on-line customers claimed that the average query was a two or three term phrase. To some extent, this explains our better performance in TREC-4 where the queries were smaller, as opposed to TREC-2.

Some ideas for future tests include the following: 1) automatic term expansion, 2) automatic term weighting (using TREC-3 and TREC-4 queries as the "ideal" for generating proper weights), 3) more sophisticated query structures derived from the semantic networks, 4) additional testing and statistical analysis. In particular, we expect to start looking at the behavior of the 50 TREC-3 queries in more detail, to see if certain types of queries behave in statistically similar ways. Our feeling is that many of our evaluations fail to show improvement because some queries get better while others get worse, and so the average performance is washed out. If we can categorize the queries into sets which perform in similar ways, this may lead to cleaner evaluations of our ranking functions and statistics.

Document Retrieval Using
The MPS Information Server
(A Report on the TREC-4 Experiment)

François Schiettecatte
Valerie Florance, Ph.D.
(francois@fsconsult.com, vf@fsconsult.com)
FS Consulting
1890 Highland Avenue, Rochester, NY, 14618
http://www.fsconsult.com

1 Introduction

This paper summarizes the results of the experiments conducted by FS Consulting as part of the Fourth Text Retrieval Experiment Conference (TREC-4). FS Consulting participated in Category C, ran the ad hoc experiments, and produced two sets of results, fsclt1 and fsclt2. Our long-term research interest is in building systems that help users find information to solve real-world problems. Our TREC-4 participation centered on three goals: to model information seeking behavior of an average searchers; to refine a retrieval system for use in a real-life setting; and to build tools to help searchers retrieve information effectively from large databases. Our two TREC-4 experiments were designed around a model of an experienced end user of information systems, one who might regularly use a system like the MPS Information Server while seeking information in a workplace or library setting. Our TREC-4 experiments provided us with baseline experience for initiating future retrieval projects and for participation in the TREC-5 interactive track.

2 Overview of FS Consulting TREC-4 Experiments

In the TREC-4 experiments we set out to answer two questions:

- How does our system perform on large databases using queries constructed by an experienced end-user/searcher?

We began with the assumption that our information seeker had previous experience using online retrieval systems commonly available in libraries (e.g., online library catalogs and bibliographic systems like MEDLINE). Although the search interface varies considerably, most library systems default to a novice-type search interface that allows a searcher to enter one or two terms and apply a Boolean operator to relate them. To aid the more advanced end-user/searcher, academic libraries typically provide search aids and training sessions that teach searchers to construct more complicated Boolean statements and employ controlled vocabularies for term selection. Additionally, information seekers who patronize librarian-mediated search services have observational experience to draw upon: often, they are asked to create query statements, or to work directly with librarians performing the search as they construct and revise search statements. It is this combination of context and experience that constitutes the background knowledge of our experienced searcher.

For our initial experiment, fsclt1, we allowed only manually-constructed queries, constructed offline. The searcher was permitted to employ any number of terms into simple or complex Boolean arguments. For this experiment, a single user query was entered for each topic, and a relevance ranked output was generated for each, using standard system features of the MPS Information Server.

- Can relevance ranking be improved within a retrieved set?

Studies and in-the-field observation of information seeking behavior demonstrate a number of important characteristics of real-world information seekers:

- they seek information to solve a problem;
- they want only enough relevant documents to answer their question;
- they don't want to review long lists of documents;
- their own relevance assessments are not absolute.

In our model of the experienced searcher, the information seeker is assumed to have an inexplicit mental model that shapes the size, shape and texture of a satisfactory search outcome. Although a system may retrieve 1000 documents in response to this searcher's query, s/he is not likely to review more than 10% of those before changing strategies, declaring the problem solved, or abandoning the search. Thus, we make the assumption that the best items must appear in the first 100 documents if they are to be viewed at all.

For our second experiment, fsclt2, we tested a tool that could help move the best documents to the top of an existing retrieved set. Reasoning that useful documents might be scattered throughout an initial retrieval set, we designed an automated query expansion feature to work within that set. For this experiment, the retrieval result set from each query in our first experiment was re-ordered using a relevance feedback algorithm.

3 Searcher Model and Guidelines

Because fsclt1 and fsclt2 employed the same query formulations, the same searcher model and guidelines apply to both. All query statements for the experiments were constructed by one person. The initial parameters of the searcher 'model' were defined as follows:

- s/he regularly searches on-line catalogs and bibliographic databases in an academic setting;
- s/he may have some search training, but is not a professional searcher;
- s/he dislikes reviewing large search outputs;
- s/he is seeking information to solve a real-life problem;
- s/he may not be a content expert in the topic area of a given question.

3.1 Instructions to Searcher

The following instructions guided query formulation:

- prepare a single search statement that will capture the most relevant documents for a given topic;
- use single or multiple terms, employing wild card capability to capture multiple versions of a word, and/or quotes around several words (e.g., "cardiac arrest") to create a fixed phrase;
- apply boolean logic as desired, using AND, OR or NOT operators. Create nested statements using parentheses if desired;
- consult the stop word list if needed, but no other databases are available for consultation;
- the total time taken to prepare a single query should not exceed 5 minutes.

3.2 Searcher Training

In preparation for the experiment, the searcher performed training exercises using the TREC-4 training data. First, general capabilities of the system and features of the search engine were described. Then, three topics (Christian coalition, prison overcrowding, oil spills) were selected from the training topics by the searcher. For each topic, a query formulation constructed by the searcher was run against the test database. Results were analyzed using the Trec Eval program. Lists of document headlines were provided to the searcher for examination. The searcher was allowed to reformulate and re-run training queries as many times as she desired. The search interface for training exercises employed a custom client application created for this experiment and MacWais, a freely available WAIS client for the Macintosh.

4 System Configuration

The MPS Information Server is a commercial full-text retrieval system that runs on a large number of Unix based platforms. Given a user query, the MPS system returns a list of relevance-ranked documents from a database. The system is capable of performing simple or complex term or phrase searching using parentheses, wildcards and Boolean operators. Soundex and fielded searching are also supported. The system is designed to favor precision over recall when performing searches. Because it supports a number of different protocols, including WAIS-88, Z39.50-V2, HTTP and Gopher+, the MPS Information Server is capable of responding to search requests from a wide variety of clients applications.

The TREC-4 experiments employed version 2.0 of the MPS Information Server running on a SparcStation 5/70. Four gigabytes of disk space were set aside, 2 GB for the TREC data and 2 GB for indices. For the purposes of the experiments, we built a custom client application. Running on an Apple Macintosh under A/UX 3.0, the custom client communicated with the MPS Information Server using the WAIS-88 protocol. This client application was designed to read TREC topic files, build a query by extracting a specific field (or fields) from the individual topic entries, run the queries against the server and save the query results in the TREC result format to a specified file. The results files could then be processed by the Trec Eval program to obtain the precision-recall values for the run.

A special parser was built to index the TREC database. We chose to extract the document ID field (the <DOCNO>) and the document title (where a title was available) to create a document headline for each document. The rest of the document was indexed as plain text, with the SGML tags extracted from the text, and the words stemmed using a plural stemmer. No additional information was extracted apart from the word positions to allow phrase and proximity searching. All keywords in the news articles were ignored, as required by the instructions.

While the MPS system's indexer starts up with an initial stop-word list (containing 377 words), it can choose to convert a word to a stop word if that word's total occurrence in the database reaches a specified value. The stop word value, which is site- and collection-dependent, would typically be set anywhere in the range of 20,000 to 50,000 occurrences. For the TREC-4 experiments, it was set at 150,000 occurrences to retain as many words as possible in the database. This resulted in a final stop-word list of 626 words.

Two databases were created, one containing disks 1 and 2 (the ad hoc training database) and the other containing disks 2 and 3 (the ad hoc test database). Each database took approximately 14 hours to build; their index sizes were about 800 MB each.

5 TREC-4 Results for FS Consulting Experiments

5.1 Experiment fsclt1

In the first experiment, the searcher created a single written query for each of 50 ad hoc topics. The queries were entered into the system and processed as a batch, without further involvement of the searcher. Search results were saved to a TREC-formatted result file.

5.1.1 Searcher Performance

Training exercises influenced the searcher's query formulation behavior in the following ways:

- she mistrusted the wild-card capability, preferring to enter multiple forms of a word;
- she was reluctant to add long lists of multiple synonyms, believing that they would dilute search results;
- she tried work-arounds to avoid the stop-list for common words like 'states' and 'united' (e.g., using "United States" as a bounded phrase).

The following examples are typical formulations used for fsclt1. The searcher wrote out the formulations, which were entered into the system by the researcher (FS) without further discussion or modification.

Topic 202: (nuclear AND treaty) AND ((violate OR violation) OR (Monitor OR monitoring))

Topic 212: ("intellectual property" OR copyright OR trademark OR patent) AND violation

Topic 217: (extraterrestrial OR extra-terrestrial) AND (life OR intelligence)

Topic 218: "steel mill" AND ("steel industry" OR (steel AND production))

Topic 221: ((drug OR gang) AND warfare) AND (prevent or prevention)

Topic 224: ("high blood pressure" OR hypertension) AND (treat OR treatment)

Topic 228: (pollution AND (reduce or reduction OR abate OR abatement)) OR (environment AND (recover OR recovery OR improve OR improvement))

Topic 231 ("National Endowment for the Arts" OR NEA) AND (appropriate OR appropriation OR budget OR fund OR funding)

Topic 241: ("professional conduct" OR malfeasance) AND (lawyer OR doctor OR physician OR attorney)

Topic 244: ("United States" OR U.S. OR America) AND Japan AND "trade balance"

Topic 248: (blind OR blindness) AND (electronic OR technology)

Most query formulations for fsclt1 employed parentheses and the AND and OR Boolean operators. As the examples indicate, not all capabilities of the system were employed (e.g., wildcard, soundex and "NOT" operator were not used for example). The bounded phrase was the most common special feature used. Missing parentheses in the examples above suggest that logic statements were not always constructed properly.

5.1.2 Server Performance for fsclt1

The results for fsclt1 produced the following precision/recall figures over all of the topics:

```
Queryid (Num):       all  fsclt1
Total number of documents over all queries
    Retrieved:    11472
    Relevant:      6501
    Rel_ret:       1103
Interpolated Recall - Precision Averages:
    at 0.00       0.5858
    at 0.10       0.3066
    at 0.20       0.2348
    at 0.30       0.2096
    at 0.40       0.1753
    at 0.50       0.1146
    at 0.60       0.0513
    at 0.70       0.0250
    at 0.80       0.0204
    at 0.90       0.0000
    at 1.00       0.0000
Average precision (non-interpolated) over all rel docs
                  0.1303
Precision:
  At    5 docs:   0.3918
  At   10 docs:   0.3571
  At   15 docs:   0.3306
  At   20 docs:   0.3122
  At   30 docs:   0.2810
```

```
    At  100 docs:   0.1524
    At  200 docs:   0.0913
    At  500 docs:   0.0425
    At 1000 docs:   0.0225
  R-Precision (precision after R (= num_rel for a query) docs
  retrieved):
      Exact:        0.1840
```

Overall, for all topics, 20% of the relevant documents were retrieved from the database and only 10% of the documents retrieved were relevant. Analysis of the distribution of relevant documents retrieved for each topic revealed that most were concentrated in the top 100 documents retrieved for each topic. Table 1 below shows R-precision scores for fsclt1's topics as compared to all TREC-4 participants.

Table 1: R-precision scores for fsclt1 at 100 and 1000 documents

R-precision Scores	@ 100	@ 1000
Above Median	15	4
At Median	4	8
Below Median	9	9
Poor	15	21
Worst	6	7
Average precision, 49 topics: .1303 • R-precision, 49 topics: .1840		

At the 100-document level, the MPS Information Server did reasonably well: 19 of the topics were at the median or above it. Topics 216,224, 226, 232, 234, 235, 242 and 245 showed 'above Median' R-precision scores at 100 documents. At 1000 documents, most of the fsclt1 results fall below the median. It should be noted, however, that only 5 out of the 49 searches retrieved 1000 documents.

Figures 1 and 2 below summarize differences in number of relevant documents retrieved for each topic in fsclt1, as compared to the overall TREC-4 median scores. Figure 1 demonstrates variation from the TREC-4 median at 100 documents; Figure 2 does the same at 1000 documents.

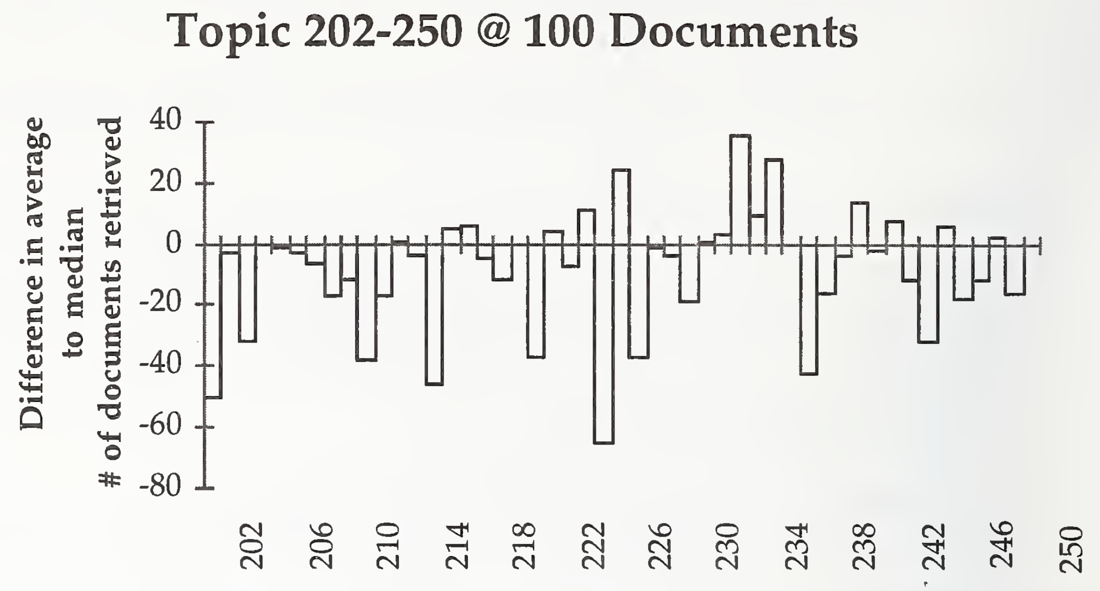


Figure 1

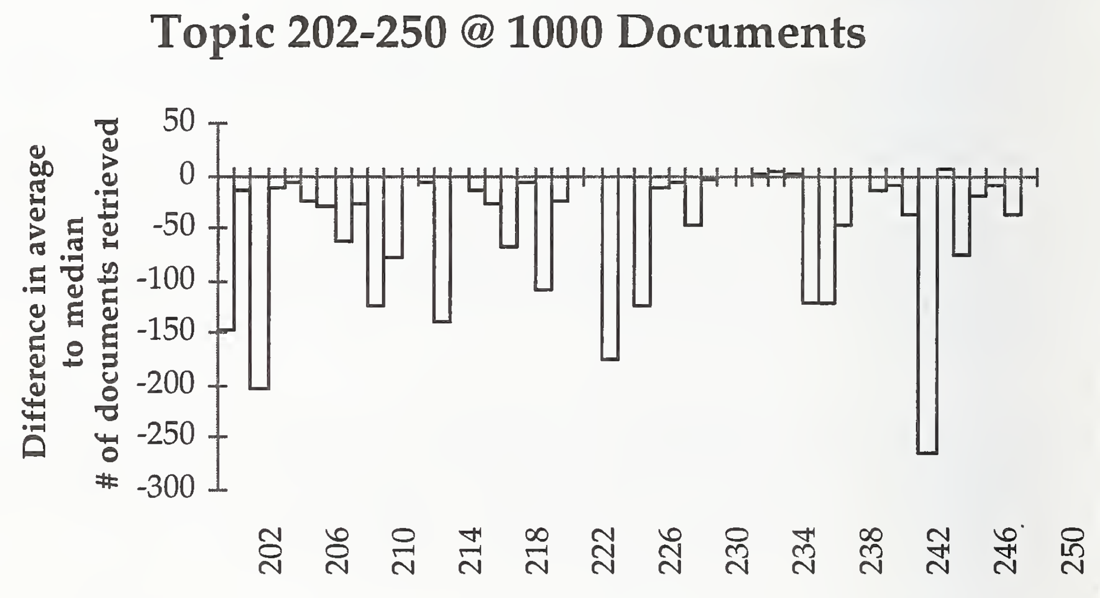
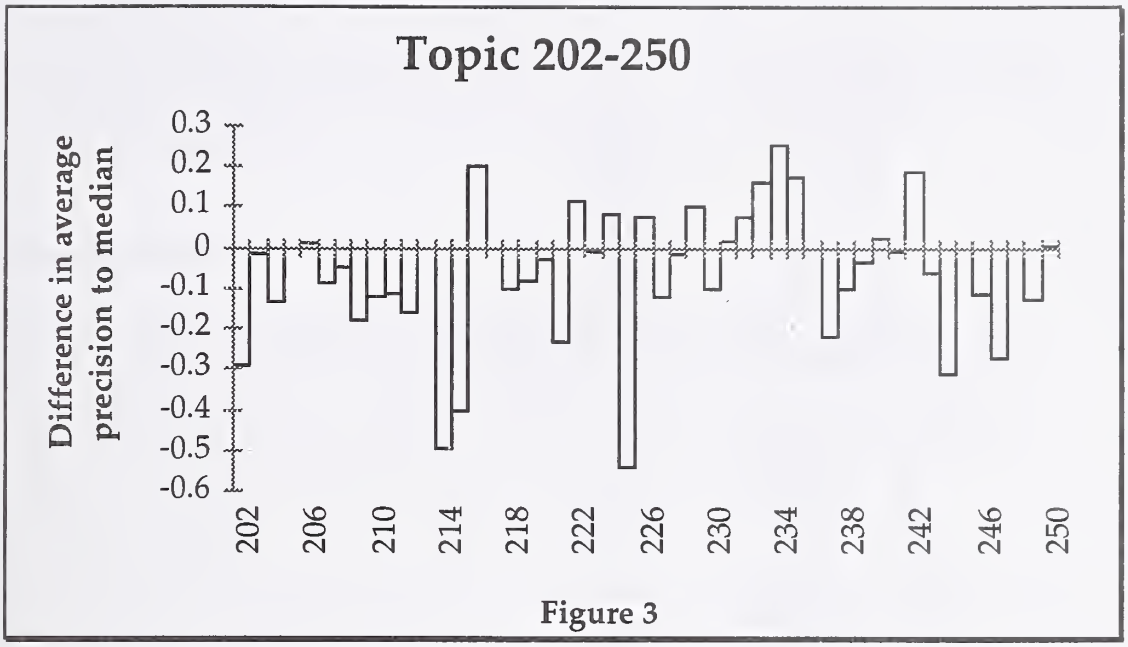


Figure 2

Figure 3 summarizes the overall variation from median R-precision scores for each topic in fsclt1.



Topic 202-250

Figure 3

5.1.3 Success/Failure Analysis for fsclt1

Examination of results from fsclt1 centered on the most and least successful topic results. One important point that emerged immediately was the effect of the search formulations on retrieval success. Successful queries (i.e. those, whose results fell at or above the TREC-4 median scores) used fewer terms and relatively simple Boolean arguments. Results are consistent with the MPS Information Server's design that favors precision over recall. When fewer than 50 relevant documents were retrieved, half of them were ranked in the first 20 documents. When more than 500 documents were retrieved for a topic, all the relevant documents were ranked in the first 100 documents

5.1.3.1 Success analysis for fsclt1

Review centered on topics for which results were at median or better when compared to the overall outcomes for all TREC-4 participants. Topic 216, on osteoporosis, was one of the most successful outcomes in fsclt1. The query was formulated as *osteoporosis AND ((prevent OR prevention) OR (treat OR treatment)).*

The precision/recall figures for this topic were as follows:

```
Queryid (Num):        216  fsclt1
Total number of documents over all queries
    Retrieved:        82
    Relevant:         36
    Rel_ret:          29
Interpolated Recall - Precision Averages:
        at 0.00       1.0000
        at 0.10       1.0000
        at 0.20       1.0000
        at 0.30       1.0000
        at 0.40       0.8333
        at 0.50       0.7200
        at 0.60       0.6875
        at 0.70       0.5417
        at 0.80       0.3973
        at 0.90       0.0000
        at 1.00       0.0000
    Average precision (non-interpolated) over all rel docs
                      0.6439
    Precision:
      At    5 docs:   1.0000
      At   10 docs:   1.0000
      At   15 docs:   0.8667
      At   20 docs:   0.7500
      At   30 docs:   0.7000
      At  100 docs:   0.2900
      At  200 docs:   0.1450
      At  500 docs:   0.0580
      At 1000 docs:   0.0290
    R-Precision (precision after R (= num_rel for a query) docs
    retrieved):
        Exact:        0.6111
```

Note that while most of the relevant documents were retrieved (29 out of 36), only 82 documents were retrieved for the whole query. This produced a small results set for the user to look at, with most of the relevant documents concentrated at the top of the result set. All of the top 10 documents were relevant, as were 13 of the top 15, 15 of the top 20, and 21 of the top 30. This topic was most successful in terms of both the experiment's goals and the stated parameters of the searcher model.

5.1.3.2 Failure Analysis for fsclt1

Topics 208, 211, 214, 215, 243, 249 had a 'worst' R-precision score at 100 documents when compared to overall TREC-4 results. Examples of query formulations for 4 of these topics follow:

Topic 211: *((DWI OR drunk) AND (drive OR driving) AND (penalties)) AND mortality*
Topic 214: *"self-induced hypnosis"*
Topic 215: *"infant mortality rate" AND ("United States" OR U.S. OR America) AND (higher OR lower)*
Topic 249: *weather AND "rain forest" AND (deplete OR destroy OR disappear)*

To explore the effect of query construction on the retrieval outcome, post TREC-4 experiments were run (see section 6.1 below) in which several queries were reformulated by dropping one or more phrases. In every case, the retrieval scores improved.

5.2 Experiment fsclt2

The second experiment centered on re-ordering relevant documents within a retrieved set. We began by running each unedited query from fsclt1 against the TREC-4 database. For each result set, the first three documents in the set were flagged as being most relevant. Up to 10 terms were selected from each document and, of those, the most 'interesting' terms were chosen for automatic query expansion. Those terms were added to the original query with an exclusive OR and assigned weights equal to 10% of those assigned to original query terms. The query was run again and the results were saved to a TREC format result file. The searcher who formulated the original queries was not involved in this experiment.

The relevance feedback algorithm employed for query expansion in this experiment works by ranking all terms in selected documents by frequency of occurrence. The ten most 'interesting' terms were chosen from that list for further use. The most 'interesting' terms were defined as neither the most frequent or infrequent terms. Rather, frequency parameters were specified to eliminate the high and low ends of the spectrum. The experiment's final result sets were produced by expanding each original query to include new terms, assigning weights to the old and new terms, and re-ordering documents in the set based on new relevance weights. In no case were additional documents added to the original set.

This automated query expansion feature was designed as a tool that could be used by information seekers who, having retrieved a large set from an initial search, wish to increase the likelihood that all relevant documents retrieved were listed in the first 30 or 40 titles in the output list.

5.2.1 Server Performance for fsclt2

Experiment 2 produced the following precision/recall figures for all the topics:

```
Queryid (Num):        all  fsclt2
Total number of documents over all queries
    Retrieved:    11472
    Relevant:      6501
    Rel_ret:       1108
Interpolated Recall - Precision Averages:
    at 0.00       0.5830
    at 0.10       0.3028
    at 0.20       0.2275
    at 0.30       0.2012
    at 0.40       0.1673
    at 0.50       0.1077
    at 0.60       0.0465
    at 0.70       0.0238
    at 0.80       0.0206
    at 0.90       0.0000
    at 1.00       0.0000
Average precision (non-interpolated) over all rel docs
                  0.1248
Precision:
  At    5 docs:   0.3755
  At   10 docs:   0.3510
  At   15 docs:   0.3238
  At   20 docs:   0.3112
  At   30 docs:   0.2796
  At  100 docs:   0.1504
  At  200 docs:   0.0888
  At  500 docs:   0.0419
  At 1000 docs:   0.0226
R-Precision (precision after R (= num_rel for a query) docs
retrieved):
    Exact:        0.1826
```

These results are not very different from fsclt1 results. Table 2 shows R-precision scores for fsclt2 topics compared to all TREC-4 participants.

Table 2: R-precision scores for fsclt2 at 100 and 1000 documents

R-precision Scores	@ 100	@ 1000
Above Median	14	4
At Median	4	7
Below Median	9	10
Poor	14	18
Worst	8	10
Average precision, 49 topics: .1248 • R-precision, 49 topics: .1826		

For the fsclt2 experiment, more relevant documents should have appeared in the first 100 documents for query topics. As this table indicates, that did not happen, and results at 1000 documents showed unpredicted repatterning.

Figures 4 and 5 below summarize differences in the number of relevant documents retrieved with respect to the TREC-4 overall median at 100 documents and at 1000 documents respectively. With few exceptions, data are identical to those presented in Figures 1 and 2.

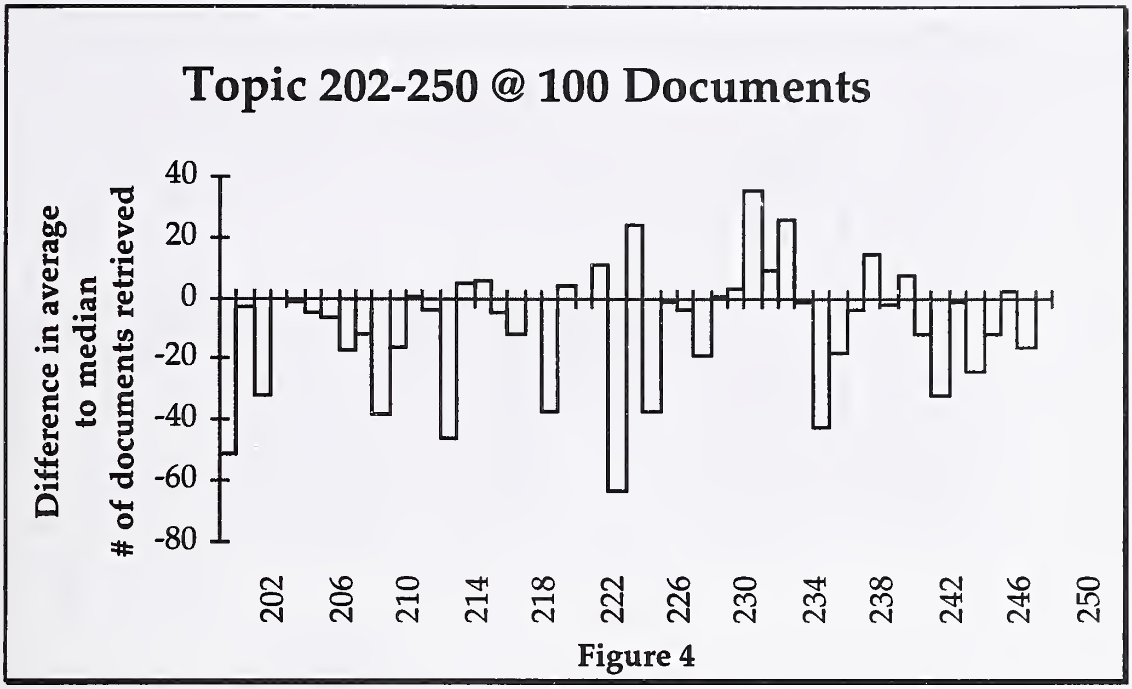


Figure 4

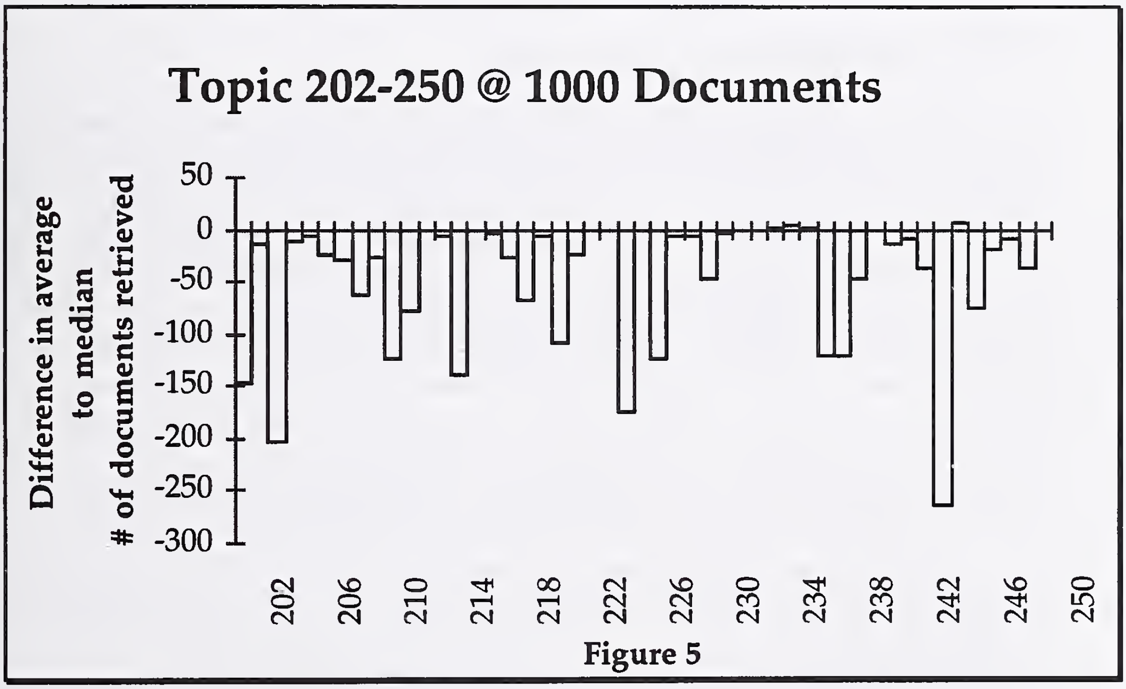


Figure 5

Figure 6 below summarizes the differences in R-precision for fsclt2 with respect to the TREC-4 median; these data are nearly identical to those in Figure 3.

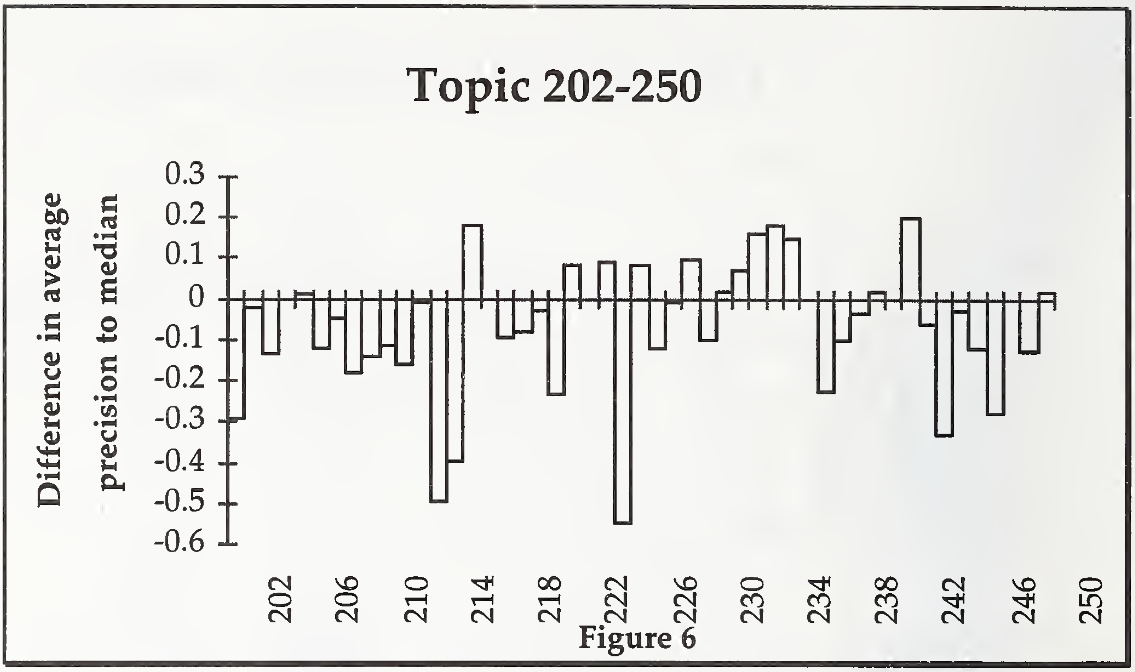


Figure 6

5.2.2 Success/Failure Analysis for fsclt2

The overall average precision and R-precision scores for fsclt2 were worse than those for fsclt 1, although topics 224, 226, 240 and 248 showed slightly-improved R-precision scores at 100 documents. In 12 cases, the average precision was improved. In 2 cases, more relevant documents appeared in the top 20 documents than had in fsclt1. Given the unsatisfactory nature of the results, we chose to ignore these few successes and concentrate on failure analysis for fsclt2 data. We began with a topic-by-topic comparison of results from the two TREC-4 experiments. By examining differences between 100 and 1000 document scores within and between experiments for each topic, 5 topics were identified for which R-precision scores dropped by 2 or 3 levels (as listed in Tables 1 and 2). Topics 240 and 248 were examined in detail.

For topic 240, the original query formulation was *terroris* AND (control OR prevent OR prevention OR combat)*. The title of the first (i.e., most relevant) document was *AP-Carter gratified by new wave of popularity....* Candidate terms available for automated query expansion were: *situations, daydream, 0950, 0921, effusive 1433edt, deporting, complimented, am-carter, ax2..* The actual expansion terms used, based on the selection parameters, were *am-carter* and *ax2..* Clearly, the terms used for query expansion were irrelevant to the topic and could not have a positive effect on set re-ordering.

For topic 248, the original query formulation was *(blind OR blindness) AND (electronic OR technology)*. The title of the first (most relevant) document was : *Pentagon joins battle....* This document was a multi-section document;

an article on river blindness appeared after the initial Pentagon piece. The ten most frequent terms were : *0326est, bissau, government-industry, burkina, faso, benin, togo, deplete, niger, ozone-damaging.* Based on the term selection algorithm, the only term chosen for query expansion was *ozone-damaging.* It is clear that the term selection algorithm was not appropriate, given a document of this type.

Review of individual topic data led to the inescapable conclusion that the relevance feedback algorithm was flawed. Upon further examination, a bug was found in the custom client created for the TREC-4 experiment. Rather than selecting the three most relevant documents for use in the automated query expansion, the system selected only the first document, employing the extracted terms three times. As would be expected, the bug's effect seriously compromised our ability to judge the usefulness of the query expansion tool and rendered the fsclt2 results only minimally useful.

6 Discussion of FS Consulting TREC-4 Results

The MPS search engine is designed to operate in an interactive setting, where quick response and high precision are generally preferable to high recall. (High recall can be achieved by creating several different queries for the same topic; this is the recommended search strategy when high recall searches are required). The TREC-4 results for fsclt1 and fsclt2 show evidence of this design decision, as most queries, even well-constructed ones, returned less than 1000 documents. Also, levels of precision/recall were generally good at 100 documents, but poor when one looked at the top 1000 document. Results indicated a need for improvement in both searcher and system domains.

6.1 Searcher improvements

Examination of the query formulations indicated that the searcher did not take full advantage of system features, and constructed queries that were not optimal for the search system. For example, multi-concept nested query statements reduced recall in many of the searches. Whether or not these formulations are typical of average searchers, it seems clear that the training period did not produce sufficient understanding of the system's strengths and weaknesses, nor were optimal query models presented and reinforced. The ad hoc track's constraint against interactive involvement of the searcher contributed to the problem, since the user could not see results of one search and modify her strategies accordingly.

6.2 System improvements

As indicated earlier, several system problems contributed to poor test results.

Flaws in the custom client application caused the automated query expansion tool to work improperly. Aside from flaws in the custom client application, the relevance feedback algorithm (i.e., the selection criteria for picking expansion terms) did not work as desired. In some cases, it worsened precision/recall scores between experiments 1 and 2. Indexing the entire document, including news message header information, introduced inappropriate search terms. This reduced efficacy of the relevance feedback algorithm and contributed to the poor results demonstrated in fsclt2.

6.3 Initial Post TREC-4 Improvements

After success and failure analysis on the TREC-4 results, we explored modifications at the searcher and system level. In both areas, slight modifications produced improved results, as outlined below.

6.3.1 Searcher modifications

Six of the original query formulations for TREC-4 topics were reformulated as indicated below, and run against the TREC-4 databases.

Topic 211: *((DWI OR drunk) AND (drive OR driving) AND (penalties)) AND mortality* became *DWI OR (drunk AND (drive OR driving) AND penalties)*
Outcome: 31 relevant documents retrieved, compared to 0 in fsclt1.

Topic 214: *"Self-induced hypnosis"* became *hypnosis*
Outcome: 5 relevant documents retrieved compared to 0 in fsclt1.

Topic 215: *"infant mortality rate" AND ("United States" OR U.S. OR America) AND (higher or lower)* became (alt. 1) *"infant mortality" AND (higher or lower)* and (alt. 2) *"infant mortality"*
Outcome: Alt.1 found 61 relevant documents and Alt.2 145 relevant documents, compared to 20 in fsclt1.

Topic 243: *"fossil fuel" AND (utilization or use)* became *"fossil fuel"*
Outcome: 36 relevant documents retrieved, compared to 0 in fsclt1.

Topic 244: *("United States" OR U.S. OR America) AND Japan AND "trade balance"* became *Japan AND (trade or balance)*
Outcome: 267 relevant documents retrieved, compared to 35 in fsclt1.

Topic 249: *weather AND "rain forest" AND (deplete or destroy OR disappear)* became (Alt.1) *"rain forest" AND (deplete OR destroy OR disappear)*; (Alt.2)*"rain forest" AND deplet* OR destroy OR disappear)* ; (Alt.3) *"rain forest" OR rainforest AND destr*;* (Alt.4) *"rain forest" OR rainforest OR rainforest*

Outcome: Alt.1 retrieved 5 relevant documents, Alt.2 retrieved 7, Alt.3 retrieved 22 and Alt.4 retrieved 35 relevant documents compared to 1 in fsclt1.

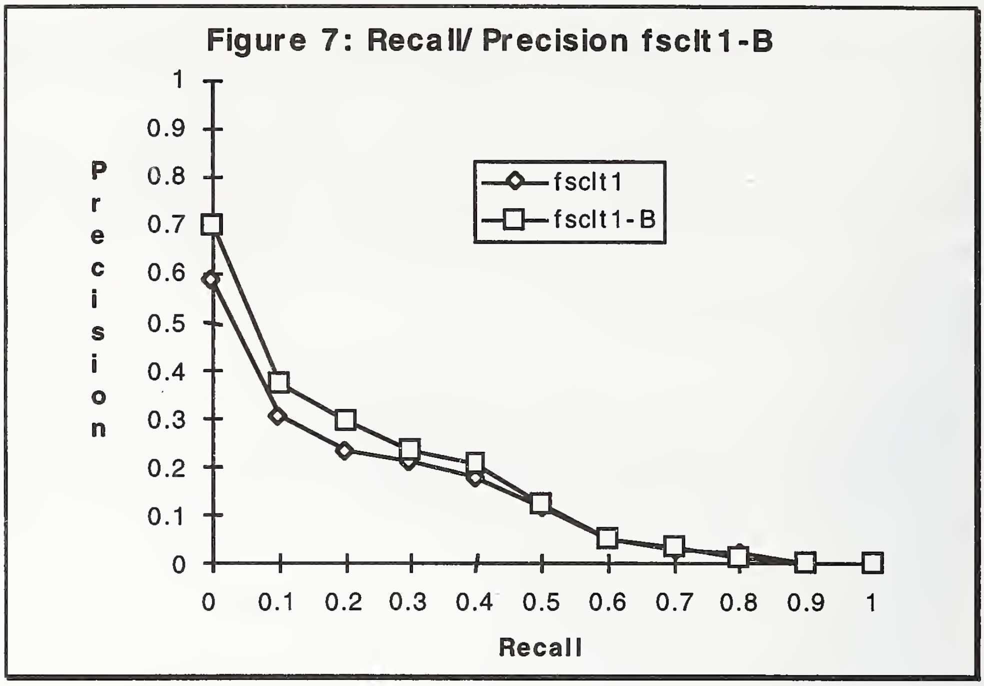
As the outcomes indicate, simplifying query formulations significantly improved retrieval results. For future experiments, training will be tuned to demonstrating outcomes of different query constructions.

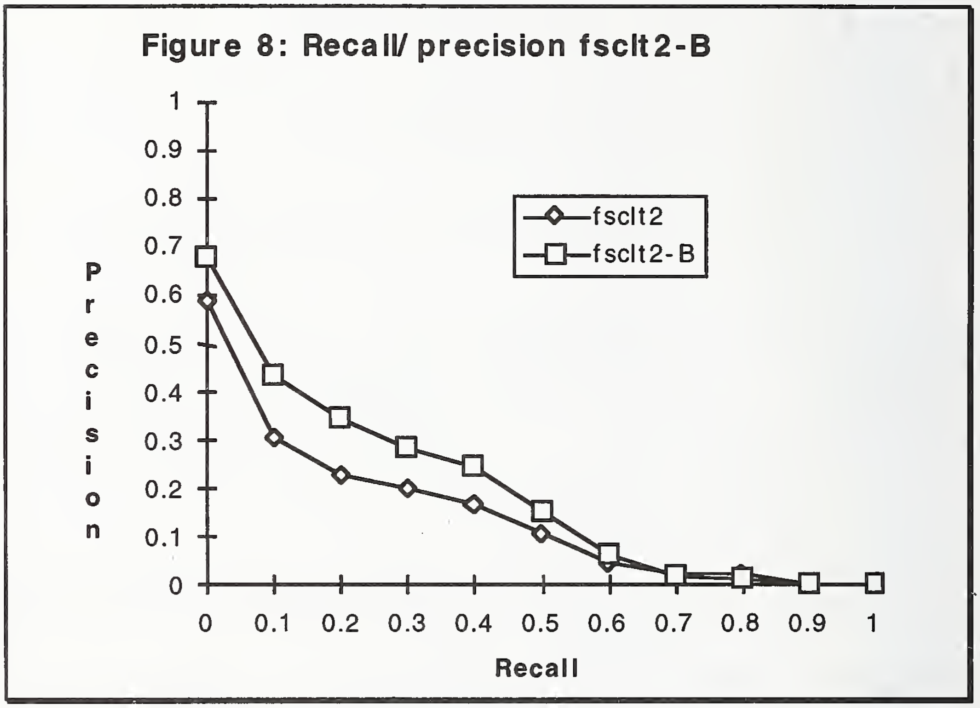
6.3.2 System modifications

Several aspects of the search mechanisms and relevance feedback algorithms were reworked. Using the original query formulations for TREC-4 ad hoc topics, the same two experiments were performed. Overall results for the post-TREC experiments:

```
Queryid (Num):      fsclt1-B
Total number of documents over all queries
    Retrieved:    11060
    Relevant:      6126
    Rel_ret:       1181
Interpolated Recall - Precision Averages:
    at 0.00       0.7018
    at 0.10       0.3695
    at 0.20       0.2946
    at 0.30       0.2329
    at 0.40       0.2020
    at 0.50       0.1189
    at 0.60       0.0472
    at 0.70       0.0308
    at 0.80       0.0207
    at 0.90       0.0113
    at 1.00       0.0000
Average precision (non-interpolated) over all rel docs
                  0.1532
Precision:
  At    5 docs:   0.4583
  At   10 docs:   0.4146
  At   15 docs:   0.3819
  At   20 docs:   0.3625
  At   30 docs:   0.3194
  At  100 docs:   0.1633
  At  200 docs:   0.0981
  At  500 docs:   0.0464
  At 1000 docs:   0.0246
R-Precision (precision after R (= num_rel for a query) docs
retrieved):
Exact:            0.2028
```

Figures 7 and 8 below compare the recall-precision results for the modified systems with results from the official fsclt1 and fsclt2 experiments.





The original results (fsclt1 and fsclt2) track each other very closely. The new test results (fsclt1-B and fsclt2-B) show improved precision in both experiments.

7 Future Work

The TREC-4 experiments provided baseline results and in a non-interactive environment and allowed exploration of possible directions for future work. Several themes emerged that will guide our research efforts in preparation for participation in TREC-5.

- The system will be tuned and improved. We will explore the effects of searching only selected fields in full-text documents. The query expansion tool will continue to be tested and revised. Additional relevance feedback algorithms will also be tested. A user interface will be constructed to allow the searcher to review results and make other decisions about search parameters.

- Additional examples of manual query formulations will be gathered and tested for TREC-4 topics, in order to build and improve the model of an 'average' searcher. Efforts will be made to gather formulations from searchers with different backgrounds (e.g., librarians, medical students, academic faculty, administrative and clerical staff). These data will be used to improve searcher training, and to suggest additional user tools.

- An interactive experiment will be designed based on improvements and searcher modeling to be undertaken this year.

Interactive TREC-4 at Georgia Tech

Aravindan Veerasamy
veerasam@cc.gatech.edu
College of Computing
801, Atlantic Drive
Georgia Institute of Technology
Atlanta, Georgia 30332-0280
Phone: 404-894-8791
Fax: 404-894-9442

Abstract

At Georgia Tech, we investigated the effectiveness of a visualization scheme for Information Retrieval systems. Displayed like a bar-graph, the visualization tool shows the distribution of query words in the set of documents retrieved in response to a query. We found that end-users use the visualization for two purposes:

- to gain specific information about individual documents – such as the distribution of different query words in that document.
- to gain aggregate information about the query result in general – such as getting a sense of the direction of the query results.

In general they used the visualization tool as much as the title and full text in the process of deciding if a document addresses the given search topic. In structured post-session interviews with searchers, we also obtained information about what the searcher liked, what was frustrating to them, and what they wanted in the system.

1 Introduction

At the TREC-4 interactive experiments at Georgia Tech, we were interested in investigating the effectiveness of a visualization scheme for IR systems that we have developed. The visualization scheme, as given in Figure 2, is intended to provide more information to the user about the query results in addition to just the title and full text. In ranked output systems, the naive end-user has little knowledge about why the system retrieved and ranked the documents in a given way in response to a free-form text query. This problem does not arise in boolean systems since there is no element of surprise in why a particular document was retrieved. The above-mentioned lack of knowledge in ranked output systems can be quite disturbing when a user is not able to get the set of documents he/she needs and does not know enough about the system to modify the query to get the documents he/she needs. It is with this in mind that we have developed a visualization scheme that shows the distribution of query words in the retrieved documents. This visual display of distribution information provides a good overview of the retrieved set of documents with respect to the free-form user query.

For TREC-4 we were interested in investigating how end-users used the visualization scheme. We were also interested in finding what aspects of the system were frustrating, what aspects they liked and what they wanted in the system. We have yet to do a thorough statistical analysis of the trace data to quantitatively determine the ways in which users with visualization tool acted different from the users without the visualization tool. What we report here is our observations of user interactions, information from structured interviews, and questionnaires.

In the next section we give a brief description of our system. Then we describe our experimental design followed by our observations as it relates to the visualization tool. Then we discuss user's frustrations, likes and wants.

2 System Description

For our study, we used the INQUERY retrieval engine from University of Massachusetts, Amherst [CCH92]. We built a simple graphical user interface on top of INQUERY using Tcl/Tk [Ous94]. There are two versions of our system – one with the visualization, and one without. In our base system, as shown in Figure 1, there are three windows: the top left window is for entering and editing the query. The titles of retrieved documents are displayed immediately below that window. Thirty titles can be displayed in one screen. One can scroll down to a maximum of 150 document titles. Mouse-clicking a title brings up the full text of that document in the window at the bottom right. By clicking the "Next Query Word" button in the full text window, one can position the full text display such that the next occurrence of query word in the document is at the top of the window.

One can save documents and mark documents for relevance feedback by clicking the "Save?" and "Rel?" buttons immediately to the left of the title in the title display window. The only operator that is allowed is the adjacency operator: A hyphen between two words specifies that the two words must appear right next to each other in the same order in a document in order for the word-combination to contribute to the retrieval of that document. There is no negation operator. Automatic stemming and stopping are performed.

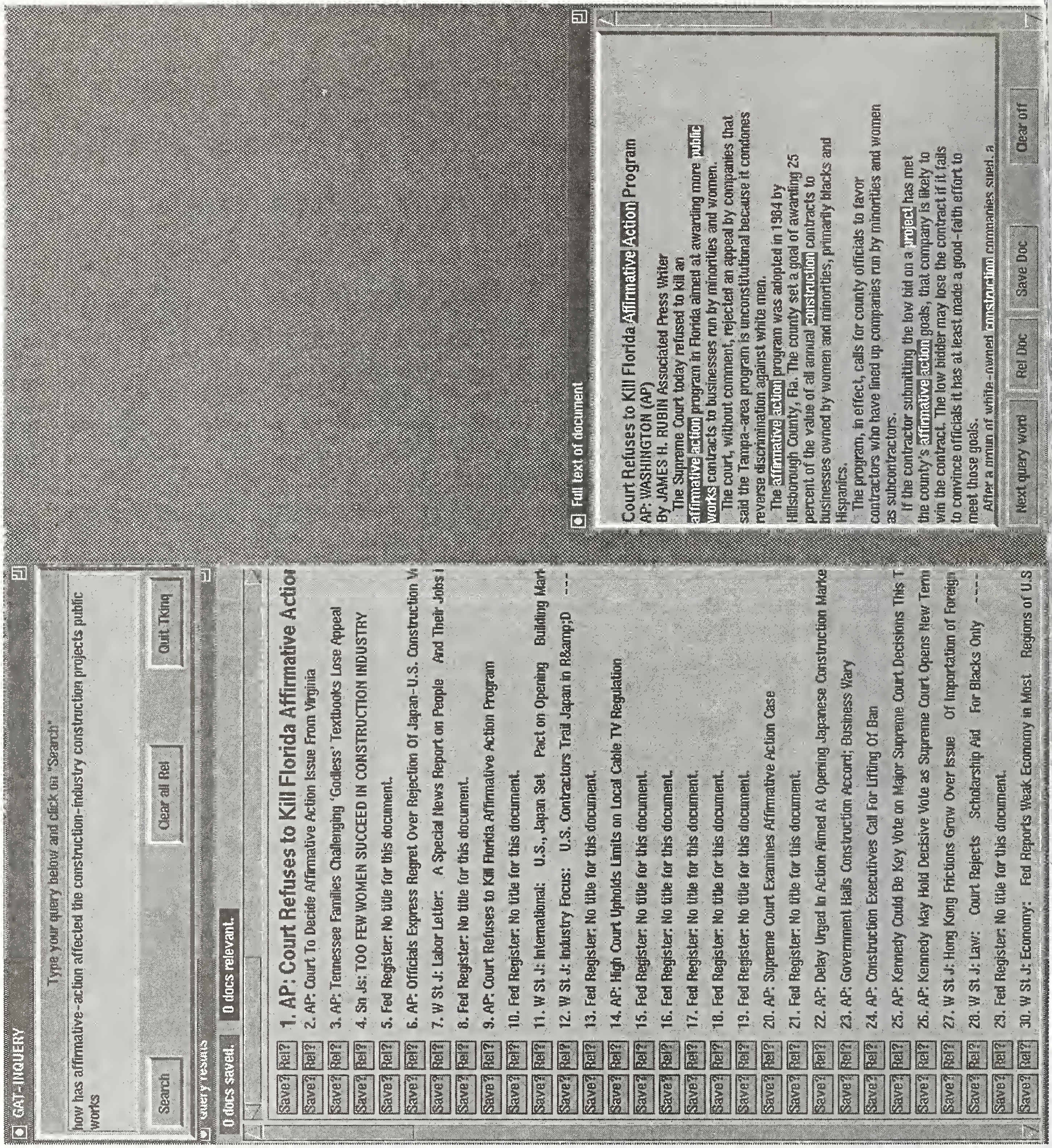
The visualization tool is displayed in another window as shown in Figure 2. It consists of a series of vertical column of bars. There is one column of bars for each document. The leftmost vertical column of bars corresponds to the document ranked 1 and the rightmost vertical column corresponds to the document ranked 150 with all the intermediate ranks lying in between. In each vertical column there are multiple bars – one each for each query word. The height of the bar at the intersection of query word row and a document column corresponds to the weight of that query word in that document. Thus if there are a handful of query words that convey the crux of the query and is very important for a document to contain these query words, one can quickly see from the visualization which retrieved documents have those important words. One can also see how many of the retrieved documents have those words in combination to get a feel for the overall goodness of query results. The effects of modifying the query, like adding a query word, would clearly be shown in the visualization. One can quickly take stock of how useful the query modification turned out. Moving the mouse cursor over the vertical columns would highlight the column directly beneath the mouse cursor and simultaneously highlight the title corresponding to that document in the title display window.

Apart from the query words typed in by the user, the visualization also shows the distribution information for words added by the system due to relevance feedback. In summary, all the words internally used by the system in computing the query results are shown in the visualization. The words in the visualization are also stopped and stemmed.

3 Experimental Setup

The searchers for our study were undergraduate student volunteers from a course on library searching at Georgia Tech. All the searchers had prior computer experience – a majority of them more than 4 years. All the students were majoring in an engineering discipline. They had differing levels of experience with the Georgia Tech Electronic Library catalog – a boolean online public access catalog.

All the users were asked to fill out a background questionnaire. They were given a tutorial on how to use the system. They were then asked to do a practice search on topic 224 for 15 minutes. Following that they were asked to find as many documents as they can that address the given information problem without too much rubbish (as specified by the interactive track guidelines). This was followed by another intermediate tutorial and then a search for a second topic. Immediately after each of the two real searches, they filled out a search evaluation questionnaire. Finally, there was a structured interview.



how has affirmative-action affected the construction-industry construction projects public works

0 docs saved. 0 docs relevant.

Save? Rel? 1. AP: Court Refuses to Kill Florida Affirmative Actior
Save? Rel? 2. AP: Court To Decide Affirmative Action Issue From Virginia
Save? Rel? 3. AP: Tennessee Families Challenging 'Godless' Textbooks Lose Appeal
Save? Rel? 4. Sn Js: TOO FEW WOMEN SUCCEED IN CONSTRUCTION INDUSTRY
Save? Rel? 5. Fed Register: No title for this document.
Save? Rel? 6. AP: Officials Express Regret Over Rejection Of Japan-U.S. Construction W
Save? Rel? 7. W St J: Labor Letter: A Special News Report on People And Their Jobs i
Save? Rel? 8. Fed Register: No title for this document.
Save? Rel? 9. AP: Court Refuses to Kill Florida Affirmative Action Program
Save? Rel? 10. Fed Register: No title for this document.
Save? Rel? 11. W St J: International: U.S., Japan Set Pact on Opening Building Mark
Save? Rel? 12. W St J: Industry Focus: U.S. Contractors Trail Japan in R&amp;D ----
Save? Rel? 13. Fed Register: No title for this document.
Save? Rel? 14. AP: High Court Upholds Limits on Local Cable TV Regulation
Save? Rel? 15. Fed Register: No title for this document.
Save? Rel? 16. Fed Register: No title for this document.
Save? Rel? 17. Fed Register: No title for this document.
Save? Rel? 18. Fed Register: No title for this document.
Save? Rel? 19. Fed Register: No title for this document.
Save? Rel? 20. AP: Supreme Court Examines Affirmative Action Case
Save? Rel? 21. Fed Register: No title for this document.
Save? Rel? 22. AP: Delay Urged In Action Aimed At Opening Japanese Construction Marke
Save? Rel? 23. AP: Government Hails Construction Accord; Business Wary
Save? Rel? 24. AP: Construction Executives Call For Lifting Of Ban
Save? Rel? 25. AP: Kennedy Could Be Key Vote on Major Supreme Court Decisions This T
Save? Rel? 26. AP: Kennedy May Hold Decisive Vote as Supreme Court Opens New Term
Save? Rel? 27. W St J: Hong Kong Frictions Grow Over Issue Of Importation of Foreign
Save? Rel? 28. W St J: Law: Court Rejects Scholarship Aid For Blacks Only ----
Save? Rel? 29. Fed Register: No title for this document.
Save? Rel? 30. W St J: Economy: Fed Reports Weak Economy in Most Regions of U.S

Court Refuses to Kill Florida Affirmative Action Program
AP: WASHINGTON (AP)
By JAMES H. RUBIN Associated Press Writer

The Supreme Court today refused to kill an affirmative action program in Florida aimed at awarding more public works contracts to businesses run by minorities and women.

The court, without comment, rejected an appeal by companies that said the Tampa-area program is unconstitutional because it condones reverse discrimination against white men.

The affirmative action program was adopted in 1984 by Hillsborough County, Fla. The county set a goal of awarding 25 percent of the value of all annual construction contracts to businesses owned by women and minorities, primarily blacks and Hispanics.

The program, in effect, calls for county officials to favor contractors who have lined up companies run by minorities and women as subcontractors.

If the contractor submitting the low bid on a project has met the county's affirmative action goals, that company is likely to win the contract. The low bidder may lose the contract if it fails to convince officials it has at least made a good-faith effort to meet those goals.

After a group of white-owned construction companies sued, a

Figure 1: Sample querying session. The window in the top-left corner is the query entry window. Immediately below that is another window where the titles of retrieved documents are displayed. To the bottom right is another window where the full text of documents are displayed.

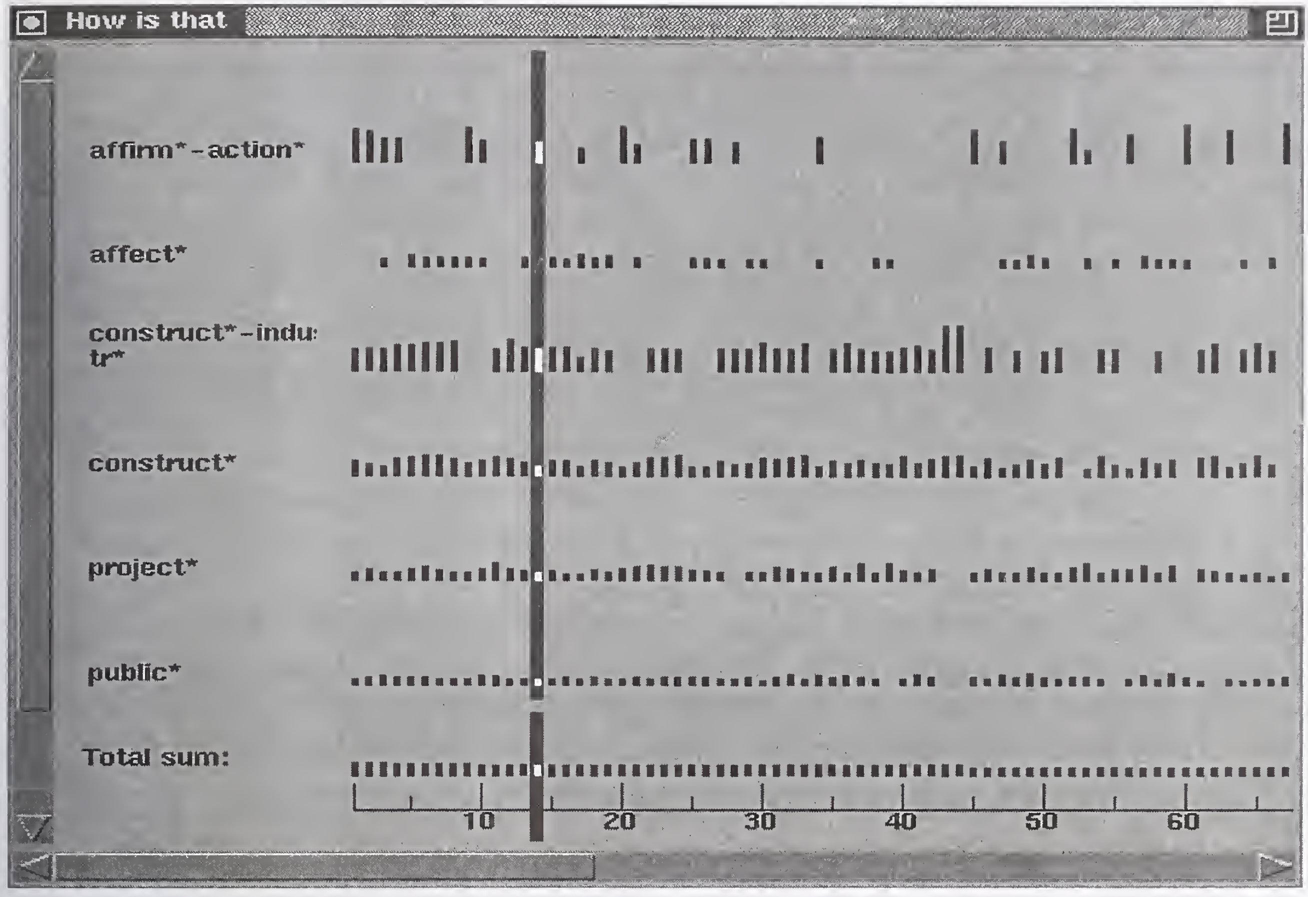


Figure 2: Visualization of results. The highlighted vertical column corresponds to document ranked 14. The title of document ranked 14 document will also be highlighted in the title display window. Clicking the highlighted vertical column brings up the full text of that document.

The searchers were divided into three groups. In each group there were 12 searchers. In the first group (hereafter named "w:w", since both first and search topics are searched WITH visualization), the searchers used the visualization tool for all the searches and the tutorial. In the second group (hereafter named "wo:w", since the first search topic is searched WITHOUT and second topic WITH visualization), the initial tutorial, the practice search and the first search was done without the visualization tool. The intermediate tutorial introduced the visualization tool and the search for the second topic was done with the visualization tool. In the third group (hereafter named "wo:wo", since both the search topics are searched WITHOUT the visualization), all the tutorials and searches were done without the visualization tool. The intermediate tutorial for the w:w and wo:wo groups was a dummy tutorial to compensate for the intermediate tutorial of the wo:w group.

Since each searcher searched for two topics and there were 12 searchers in each group, all the 24 topics were covered by each of the three groups. The 24 topics were randomly divided into 12 pairs and each pair was searched by 3 searchers, one each from the w:w, wo:w and wo:wo groups. The idea was to compare the performance among the three groups to find out the effects of the visualization scheme. Only 24 of the 25 topics for the interactive track were given to end-users in the study. The remaining one topic (topic 223) was searched by the author using the visualization tool.

The searchers were asked to think aloud as they used the system. For the most part, there was an observer in the same room using a different computer and simultaneously observing the searcher. Based on such observations while the user session was in progress, we felt that huge searcher differences in interpreting the query combined with huge differences in the nature of the search topics will greatly confound the effects of the visualization tool. As a result, we decided to run a second study. In the second study, we picked topic 242 for the practice search and the practice search was extended to 30 minutes. The intermediate tutorial was removed. We picked topics 203 and 236 for all the searchers. There were two groups of searchers for the second study – the first group had the visualization tool and second group did not have the visualization tool. There were 18 searchers in each group. By keeping the two search topics constant for all these searchers, we expected to eliminate the effects of search topic difference. It turns out that the searcher variability in interpreting the search topic is so huge among searchers that it is not fair to compare different searchers using different systems unless the search topic is extremely clear and specific.

4 End-users view of the visualization tool

A vast majority of the users mentioned visualization as one of the aspects of the system that they liked. They mentioned using the visualization tools in the following ways.

- Some searchers mentioned using it to see the importance of query words in the retrieved documents – as given by the height of the bar. They mentioned that they were more likely to look at the full text of a document if it has a higher concentration of the important query words.

- Most of the searchers mentioned using it most frequently to see the co-occurrence of important query words in the retrieved documents. They mentioned it being easier to use the visualization tool to look for the co-occurrence information than going through the full text of documents in search of occurrences of the important query words.

- Many searchers felt that the visualization in conjunction with the document title gives a fairly good idea of what the document is about. If the title looks promising and the visualization shows that the document has the right combination of query words, one is tempted to look at the full text of the document.

- They mentioned using it to get a quick overview of the number of retrieved documents a query words appears in. They mentioned using it as a checkpoint to see if a query has turned out the way they had expected it to. If not, they were tempted to readjust the query to get a better result. This happens often when some of the crucial query words are not well represented in the retrieved documents. In that case, one is tempted to add synonyms or words related to those crucial query concepts.

- Some of the searchers mentioned that the visual nature of the distribution information was much easier to identify things than reading text information. This suggests that the mental effort of reading textual information as being much higher than interpreting a simpler visual pattern, and given a choice, the users are more likely to choose the latter.

- **Disadvantages:** A few searchers mentioned that relying heavily on the visualization can also hurt as follows: They mentioned that using the bar-graph to pick out a document containing certain query words may not be indicative of the content of the document – just as the title may not be a good indicator of content. An exemplar case is searcher 35 on the topic of "status of nuclear proliferation treaties". Since almost all of the retrieved documents had something to do with "nuclear proliferation", the searcher mentioned using the visualization tool to pick those documents containing the query word "status" – only to see that the usage of "status" in the document was not in the context of nuclear proliferation treaties. Then the searcher started paying little emphasis on the presence of "status" in documents. Although relying on that information was initially detrimental, one tends to learn when and how to rely on the visualization. We believe that the presence in retrieved documents of adjectives, adverbs and verbs from the query may not be good content indicators especially when they have a high collection frequency. And relying on the visualization to select documents that have these adjectives, adverbs and verbs from the query may not help.

In summary, the visualization tool seems to help in the following ways:

- to gain more information about specific documents in addition to the title before looking at the full text. Higher concentration of important query words in a document suggests a closer look at the document.
- to gain aggregate information about the query result. The absence of important query words in a vast majority of the retrieved documents suggests query reformulation by adding synonyms and other related concepts.

5 Likes, Frustrations and Wants of users

Apart from the visualization, we were also interested in finding if there are any specific facilities that the users wanted, what features they liked, and what aspects were frustrating. While interpreting the following, we wish to reiterate that all the searchers had some amount of experience with the Georgia Tech Electronic Library catalog which is a character-based-command-driven interface to a boolean system. Some of the features they liked may arise out of the fact that they have had little experience with ranked output systems and the only other major information retrieval system they know is a character based interface to a boolean system.

5.1 Likes

- A vast majority of the searchers with the visualization mentioned that the visualization tool and relevance feedback as the two major aspects of the system they liked. Searchers without the visualization mentioned relevance feedback as the most important feature they liked.

- A number of searchers found the fact that all the information (like the user query, titles of documents and the document full text) is displayed simultaneously in one screen to be very useful. In the Georgia Tech library system, one has to switch between screens to get different types of information. There seems to be a significant mental overload in the context switch between screens. Having simultaneous access to all information seems to bring about a rich interplay between the different sources of information.

- many searchers mentioned that the mouse-based graphical nature of the interface is a significant improvement over a command line based interface.

- many searchers also mentioned that the free-form textual queries without having to worry about any syntax leads to a free flow of thought. "I like the fact that I can type in whatever comes to my mind ... knowing that it will ignore all the junk words like a, an, the, etc...".

- The "Next Query Word" feature was also liked by many searchers. They liked it because they did not have to scroll through a long document looking for occurrences of query words. (All the occurrences of query words in a document are highlighted by the system).

5.2 Frustrations

- A number of searchers mentioned that it was frustrating when the system takes a long time to get the full text of a large document. Similarly, they were also frustrated when it takes a long time to evaluate a query with a large number of relevance feedback documents. The longest delay for evaluating a query was about 2 minutes (when there are about 30 relevant documents). Most of the query evaluations took less than 20 seconds. They said that they understand that the system has to process a lot information (when there a number of relevant documents), but it was frustrating nevertheless.

- Some searchers said that it was frustrating to spend some time reading through the full text of a document and when they are halfway, realizing that they had already seen the same/similar document.

- While some searchers seemed to like having access to 150 retrieved documents, some others mentioned that 150 documents is too much especially when most of the 150 are not relevant. They seem to have the opinion that if some documents are definitely not relevant to the query, then they should not be shown. Thus, this problem is not alleviated even if one reduces the number of documents displayed. They seem to be quite sensitive about precision. They are not as sensitive about recall – since they are usually satisfied if they get a few documents concerning the topic. Based on our observations, we believe that when the non-relevant documents consistently come from a particular subject area, and when the user is not in a position to remove those documents, they tend to get more frustrated. Using subject classification schemes (where available) to negate disinteresting subject areas would help in this regard.

- In our system, when the title for a document is not available, the message "No title for this document" is displayed instead of the title in the title display window. Many of the federal register documents do not have a title and this is quite annoying to some searchers since they do not have any idea about the document content. This makes it difficult to decide whether to request the full text or not. In cases where the full text is requested, the document happens to be large and hence takes a lot of time to retrieve, thereby adding to the frustration.
- Some federal register documents do not have anything worthwhile – they consist of a listing of subject areas or table of contents. Some searchers wondered why these documents were in the database in the first place.
- Some searchers mentioned a general dislike towards federal register documents partly because they felt that many of them did not have any important piece of information, partly because in general they have no title, partly because it took too long to retrieve them.
- Some searchers were frustrated when a document that they know as non-relevant keeps coming up in the query result. The fact that they were not able to delete the document from the display seemed to add to the frustration.

5.3 Wants

Many of the frustrations mentioned above seemed to directly translate into wants for removing the causes of frustration. In addition to those wants, we observed the following:

- Many searchers expressed a desire to remove certain query words that were added by the system from relevance feedback documents – especially when they are proper names and when they are not necessarily what they are looking for.

- A number of searchers wanted a keyboard equivalent of mouse actions. This is not to say that they did not want mouse actions. It seems to be a significant effort for these searchers to move the right hand out of the keyboard, reach over to the mouse, look at the screen to position the mouse cursor, click the mouse button and move back to the keyboard.

- When asked if they felt a need to have access to an online thesaurus, some searchers expressed a desire for it and some did not. Some of those who did not want a thesaurus mentioned that relevance feedback seemed to alleviate the need for a thesaurus.

- Many searchers wanted to be able to specify that the system should definitely avoid retrieving certain documents in subsequent query iterations. They wanted to have a negative relevance feedback where the system avoids all documents like a particular nonrelevant document.

6 Acknowledgments

The tremendous help from Prof. Nick Belkin regarding the experimental setup and questionnaire design is highly appreciated. Many thanks to Prof. Scott Hudson and Prof. Shamkant Navathe who were instrumental at every stage of the interface development. Special thanks to Prof. Bruce Croft for letting us use the INQUERY retrieval system. Support from the ARPA contract No. F33615-93-1-1338 is appreciated.

References

[CCH92] J.P. Callan, W.B. Croft, and S.M. Harding. The inquery retrieval system. In *Third International Conference on Database and Expert Systems Applications*, September 1992.

[Ous94] John K. Ousterhout. *Tcl and the Tk Toolkit.* Addison-Wesley, 1994.

Improving Accuracy and Run-Time Performance for TREC-4

David A. Grossman
Office of Information Technology
3E09 Plaza B
Washington, DC
dgrossm1@mason1.gmu.edu

David O. Holmes
AT&T Global Information Solutions
Rockville, MD
David.Holmes@washingtondc.attgis.com

Ophir Frieder*
Department of Computer Science
George Mason University
Fairfax, VA
ophir@cs.gmu.edu

Matthew D. Nguyen
Coherent Design Systems
Centreville, Virginia
mnguye6@osf1.gmu.edu

Christopher E. Kingsbury
Hughes Applied Information Systems
Landover, Maryland
ckingsbu@osf1.gmu.edu

Abstract

For TREC-4, we enhanced our existing prototype that implements relevance ranking using the AT&T DBC-1012 Model 4 parallel database machine to support the entire document collection. Additionally, we developed a special purpose IR prototype to test a new index compression algorithm and to provide performance comparisons to the relational approach.

We submitted official results for both automatic and manual adhoc queries for the entire 2GB English collection and the provided Spanish collection. Additionally, we submitted results using n-grams to process the corrupted data. In addition to implementing the vector-space model, we experimented with query reduction based on term frequency. Query reduction was shown to result in dramatically improved run-time performance and, in many cases, resulted in little or no degradation of precision/recall.

1 Introduction

For TREC-4, we implemented relevance ranking queries using SQL on an AT&T DBC-1012 (formerly Teradata) parallel database machine [1]. Additionally, we implemented a special purpose IR prototype to test a new index compression algorithm and to provide performance comparisons to the relational approach.

We submitted official results for the entire 2GB English collection, both for automatic and manual adhoc queries and against the Spanish collection. We also submitted results using n-grams to process the corrupted data.

*This work supported in part by the National Science Foundation under contract number IRI-9357785.

In addition to implementing the vector-space model, we experimented with query reduction based on term frequency. Query reduction was shown to dramatically affect run-time performance without losing a significant amount of precision/recall.

We will briefly describe the implementation of our relational prototype and our special-purpose prototype in Section 2. More detailed descriptions are found in [9, 12]. Sections 3 and 5 will describe the results obtained for our English and Spanish submissions. Section 4 describes our corrupted data results, and Section 6 contains our conclusions and future work.

2 Implementation of the Inverted Index

We developed two separate implementations, a parallel relational approach and a special purpose IR approach.

2.1 A Parallel DBMS Approach to IR

Our approach treats the problem as an application of a relational database system. While parallel implementations of relational database systems are common, parallel implementations of information retrieval (IR) systems are rare. Implementing information retrieval as a relational database application provides a portable, parallel means of implementing information retrieval algorithms.

We model an inverted index with a relation DOC_TERM(*doc_id, term, tf*). A relation, QUERY (*query, term, tf*), indicates the terms in the query and their frequency in the query. DOC (*doc_id, doc_name, doc_weight*) contains the document name and the normalized weight for each document. QUERY_WEIGHT (*query, query_weight*) contains the normalized query weight for each query. Finally, an IDF(*term,idf*) relation stores the inverse document frequency for each term.

Given these relations, the following DBC-1012 SQL will compute a cosine similarity coefficient for a given query: *query_number*:

Ex: 1

```
SELECT a.query, c.doc_name, SUM(a.tf * b.tf * e.idf * e.idf) / SQRT(d.query_weight * c.doc_weight)
  FROM QUERY a, DOC_TERM b, DOC c, QUERY_WEIGHT d, IDF e
 WHERE a.term = b.term AND
       a.term = e.term AND
       b.docid = c.docid AND
       a.query = d.query AND
       a.query = query_number
 GROUP BY a.query, c.doc_name, d.query_weight, c.doc_weight
 ORDER BY 2 DESC
```

For additional details the reader is referred to [7]. The query in Example 1 is of fixed syntactic length. We implemented the cosine query on the AT&T parallel database machine. We implemented the simple dot product on *Microsoft SQL Server V4.2, Sybase SQL Server System 10,* and *Oracle 7.* Practical experimentation shows that this query performs well on each of these systems.

2.2 Special Purpose IR Prototoype

We developed a special-purpose IR system to test a new index compression algorithm. Additionally, we wanted to learn more about the implementation details of an IR system. Our system implements relevance ranking using the popular vector-space model [14].

Although compression of text has been extensively studied [2, 8], we have found little work in the area of compression of inverted indexes. Linoff and Stanfill, the most recent published work on the compression of inverted indexes, combine three techniques to compress their index [11]. We implemented this algorithm, referred to as NS compression, as well as a new algorithm, byte-aligned (BA), and compared the differences. Run-time performance for index creation and compression ratio of both TREC-3 and TREC-4 data are given below:

Run Time Performance (index construction)

Compression Type	TREC-3	TREC-4
none	3:48:47	2:38:59
ns	3:50:58	3:14:47
ba	2:56:42	3:06:58

Compression Ratio

Compression Type	TREC-3	TREC-4
none	.466 : 1	.457 : 1
ns	.128 : 1	.121 : 1
ba	.163 : 1	.157 : 1

We have found that the BA compression algorithm results in faster creation of the inverted index, but yields slightly less compression than NS compression.

Query run-time performance is give in Figure 1. Note that run-time is very slightly improved with BA. All performance numbers given here were obtained when running Sun Solaris 2.4 on an 18 processor SUN Sparc 2000. The data was spread across the processors, but the algorithms implemented were sequential.

3 English Results

3.1 Automatic

We submitted both manual and automatic results for the Category A data. All results for Category A data were submitted using the relational IR prototype. Each section of the corpus was loaded into a corresponding relation, and a larger query to UNION all the different relations was implemented.

In addition to simply loading terms, we also loaded phrases which were recognized with a crude phrase parser. A phrase was defined as a two-term sequence that did not contain a punctuation mark or a stop word.

The topics were parsed in the same fashion and both terms and phrases were incorporated into the queries. Phrase inverse document frequency (IDF) was computed as if the phrase was a single term. All terms other than stop words were used in the query.

3.2 Manual

Our automatic queries returned a union of all documents that contained one or more terms related to a query. The manual implementation restricted answer sets, in many cases, to an intersection of documents.

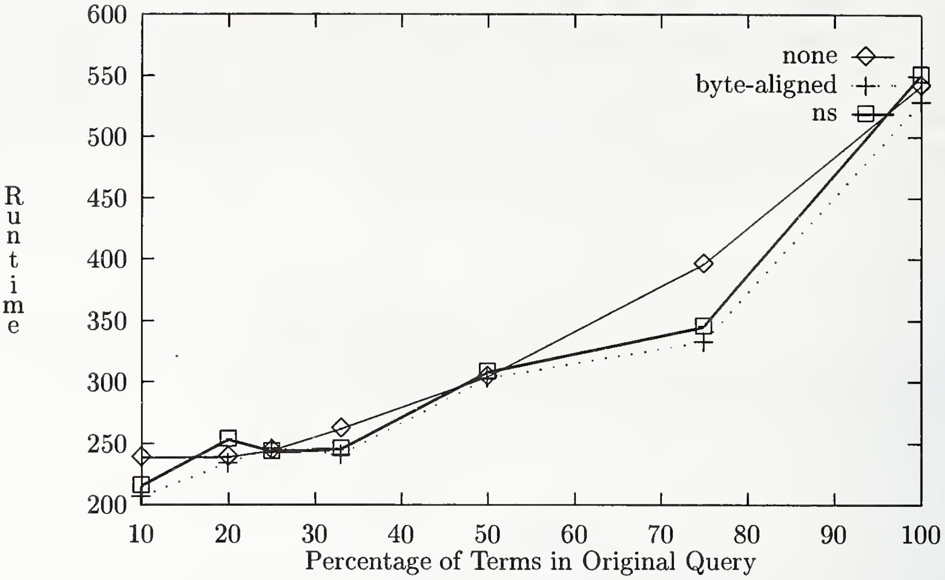


Figure 1: Query Performance (TREC-4, 2GB)

The manual queries were generated after careful study of the query terms. Related terms were placed in at most three sets, and the SQL was generated such that a document was only ranked if it contained at least one term from each set. An optional fourth set contained terms which were not required, but increased the relevance of a document if they were present. Many sets contained lexical variants of the term as we did not perform any stemming. Each set roughly corresponded to a concept.

For example, on topic 203, our answer set was an intersection of two lists. The first set contained the terms {*recycle, recycled, recycling*}. The second set contained the term {*tires*}. Qualifying documents mentioned one of the three forms of *recycle* and referenced *tires.*

To increase the relevance of documents about the economic benefit of recycling tires, we included terms and phrases such as *car tires*, *automobile tires*, *cost effective* and *profitable* to the optional fourth set. The inclusion of a document into the answer set was not dependent upon these extra terms and phrases, however, their presence affected the ranking of the document within the answer set.

Finally, world knowledge was used to generate query terms. Topic 205 requests information about paramilitary activity in the United States. For this topic, one set contained {*militia, militias*} and another set was {*Idaho, Oklahoma*} in an effort to ensure that certain states in which the militia have been active were searched.

3.3 Results

Our calibration against TREC-3 data indicated that a threshold of 100 provides the best results for English data. Hence, we used a threshold of 100 for our official results. The hypothesis for the parallel approach was that queries would run in a balanced fashion; that is, the workload for each processor would be approximately equal. This balance is critical if the approach is to be truly scalable. Table 1 indicates the amount of Processor Load Imbalance (PLIB) for CPU and DISK I/O time: $\left(\frac{max-min}{min}\right)$ measured at each query threshold. For all workloads, the processors are fifteen

Table 1: Percent of Processor Imbalance

threshold	*CPU Time*	*Disk I/O)*
10	15.0	1.5
20	12.0	1.5
25	10.4	1.2
33	9.4	0.9
50	2.0	0.7

percent or less out of balance. Given that the workload is balanced evenly among the existing processors, if processors are added, response time will be reduced. Due to resource limitations, it was not possible to empirically validate this scalability hypothesis.

Our overall results for English are given below:

	Avg. Precision	Above Median	Below Median	Equal Median
Automatic	.1385	10	37	2
Manual	.2216	25	22	2

4 Corrupted Data Results

For corrupted data, we tested the use of n-grams and the effect of reducing the number of n-grams in the query based on an automatically generated *query threshold*. All results for corrupted data were obtained using our special-purpose IR prototype.

N-grams have been used for detection of spelling errors [13, 15, 19] and text compression [16]. A survey of text compression and spelling checking may be found in [18]. More recently, n-grams have been used to determine the authorship of documents [10]. Yochum has shown n-grams are effective in implementing a high-speed document routing system [17].

The first similarity measures based on n-grams were done by Ray D'Amore and Clinton Mah in the early 1980's [6]. More recently, at TREC-3, Damashek expanded on D'Amore and Mah's work by implementing a five-gram based measure of relevance [5]. Damashek's algorithm is entirely language independent and relies upon the vector space model but computes relevance using centroid vectors.

Also at TREC-3, Cavnar implemented a vector-space approach using n-grams [4]. The effectiveness of his approach and the claim that n-grams should have resilience to errors resulted in our decision to use n-grams as part of the corrupted data track for TREC-4.

In TREC-3, we illustrated the effectiveness of automatically reducing query size based on term frequency. First, the terms in the query are sorted and only a percentage of these terms are used. We found that for higher thresholds, relatively useless terms are added to the query and there is little, if any, improvement in precision and recall.

We trained for the real data in TREC-4 (both WSJ data on disk 2 and SJMN data on disk 3) by implementing various n-grams and query thresholds against only the WSJ data on disk 2 and running the TREC-3 queries against this data. We tried n-grams of both size three and four and thresholds of 10, 20, 25, 33, 50, 75, and 100 percent of the original query. With trigrams, we obtained between 700 and 800 relevant documents for the various thresholds (increasing for thresholds from 10 to 33 and decreasing for 50 to 100). With 4-grams we found between 800 and 900 relevant documents with the threshold value increasing up to 75 and then decreasing. Hence, we selected a query threshold of 75 and 4-grams for our official results.

Our results for corrupted data are given below:

	Avg. Precision	Above Median	Below Median	Equal Median
Baseline	.1401	11	19	0
10 Percent	.1153	23	26	0

5 Spanish Results

For the Spanish data, we used the special purpose IR system to obtain our automatic results, and the same method that was used for the English queries was implemented to obtain the manual results.

5.1 Automatic Results

We developed a Spanish stop word list by identifying the top 500 most frequent terms and asking a Spanish linguist to determine which ones were really not so common across the language that they should be in a stop list. We calibrated our approach using the first twenty-five topics and the relevance assessments from TREC-3. Results for trigrams, 4-grams, and 5-grams at thresholds of 10, 20, 25, 33, 50, 75, and 100 are given in Figure 2.

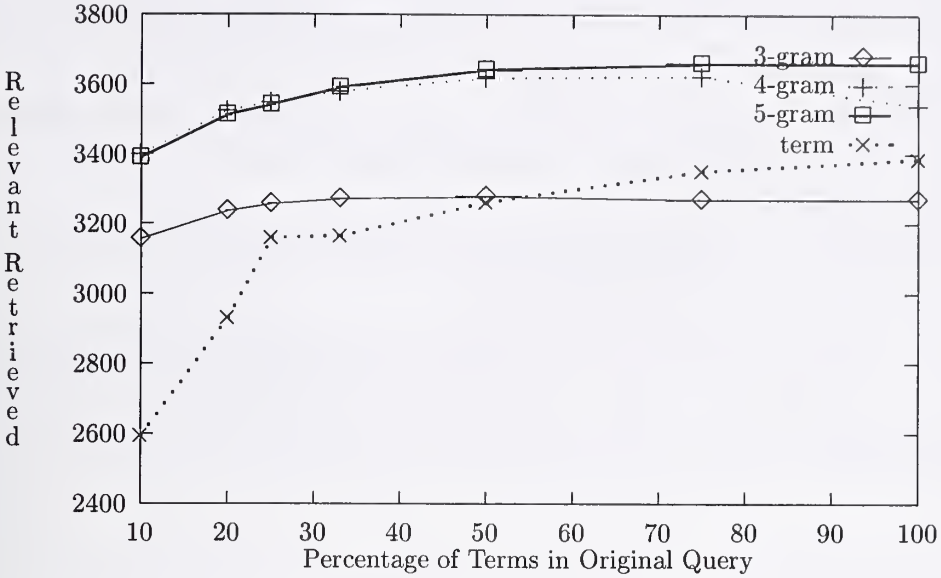


Figure 2: Spanish Relevant Retrieved

We found that our best results occurred with a threshold of 100 using 5-grams. We submitted official results using these parameters.

5.2 Manual Results

After having the queries translated by a Spanish linguist and key terms identified, we grouped the terms in sets and ran the same type of manual queries as done for the English data.

5.3 Results

Our results for Spanish data are given below:

	Avg. Precision	Above Median	Below Median	Equal Median
Manual	.1708	12	11	2
Automatic	.1722	14	6	5

Interestingly, the automatically generated queries performed better than the manual queries. The manual queries used terms, while the automatic used n-grams. The use of n-grams appears to provide a type of automatic stemming as frequently occuring prefixes and suffixes are ignored. Additionally, our lack of Spanish language knowledge made it difficult to develop sophisticated manual queries.

6 Conclusions and Future Work

Given that this was our first year as a Category A participant, we see much room for improvement. Although about half of our queries for both corrupted data and the manual English data were over the median of all participants, half were not.

Our manual queries surprised us with their accuracy. Clearly our automated system needs more improvements, such as passages and relevance feedback. With these improvements, it is likely that our manual queries will improve as well. We are working on enhancing the prototype to expand the manual queries based on relevance feedback.

For corrupted data, we were pleased that the n-gram approach showed resilience to errors. More queries for corrupted data were above the median than below. We will continue testing on the 20 percent corrupted data.

It was interesting that an n-gram approach was calibrated as superior to a term-based approach for Spanish. We look forward to official results to determine whether or not the effects we observed during calibration occurred on the TREC-4 data.

Overall, we were much improved from TREC-3. During TREC-3, we only used the category B data and our precision/recall was a mediocre .0860 and .1356, respectively. Only one of our fifty queries was above the median. Our worst results this year were vastly superior to those submitted last year (our lowest number above the median is ten). We will try to keep up this pace and improve as much in TREC-5.

References

[1] AT&T Global Information Systems. *Teradata DBC-1012 Concepts and Facilities*, March 1992.

[2] T.C. Bell, J.G. Cleary, and I.H. Witten. *Text Compression.* Prentice Hall, Englewood Cliffs, NJ, 1990.

[3] D. Blair. An extended relational retrieval model. *Information Processing and Management*, 1988.

[4] W. Cavnar. N-grams in trec-3. In *Proceedings of the Third Annual Text REtrieval and Evaluation Conference*, 1994.

[5] M. Damashek. Gauging similarity via n-grams: Text sorting, categorization, and retrieval in any language. Submitted to Science, 1994.

[6] R. D'Amore and C. Mah. One-time complete indexing of text: Theory and practice. *8th International Conferencee on Research and Development in Information Retrieval*, pages 155–164, 1985.

[7] D. Grossman, O. Frieder, D. Holmes, and D. Roberts. Integrating structured data and text: A relational approach. To appear in the Journal of the American Society of Information Science, 1995.

[8] P.C. Gutmann and T.C. Bell. A hybrid approach to text compression. In *Proceedings of the Data Compression Conference DCC '94*, 1994.

[9] D. Grossman O. Frieder D. Holmes and D. Roberts. Integrating structured data and text: A relational approach. Submitted to the Journal of the American Society for Information Science, August 1995.

[10] B. Kjell, A. Woods, and O. Frieder. Discrimination of authorship using visualization. *Information Processing and Management*, 30(1):141–150, January 1994.

[11] G. Linoff and C. Stanfill. Compression of indexes with full positional information in very large text databases. *Proceedings of the Sixteenth Annual ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 88–95, 1993.

[12] D. Grossman O. Frieder M. Nguyen and C. Kingsbury. Implementation of an information retrieval system. To be Submitted to Software Practice and Engineering, October, 1995.

[13] J. Pollock and A. Zamora. Automatic spelling correction in scientific and scholarly text. *Communications of the ACM*, 27(4):358–358, April 1984.

[14] G. Salton, C.S. Yang, and A. Wong. A vector-space model for information retrieval. *Communications of the ACM*, 18, 1975.

[15] Lars Erik Thorelli. Automatic correction of errors in text. *BIT*, 2:45–62, 1962.

[16] E.J. Yannakoudakis, P. Goayal, and J.A. Huggill. The generation and use of text fragments for data compression. *Information Processing and Management*, 18(1):15–21, 1982.

[17] J. Yochum. A high-speed text scanning algorithm utilizing least frequent trigrahs. *IEEE Proceedings on New Directions in Computing Symposium*, pages 114–121, 1985.

[18] E.M. Zamora. Survey of spelling correcting methods. *ACM Computing Surveys*, 1983.

[19] E.M. Zamora, J.J. Pollock, and A. Zamora. The use of trigram analysis for spelling error detection. *Information Processing and Management*, 17(6):305–316, 1981.

Using CONVECTIS, A Context Vector-Based Indexing System for TREC-4

Joel L. Carleton
William R Caid
Robert V. Sasseen
HNC Software Inc., 5930 Cornerstone Court West, San Diego CA 92121 USA

Abstract

This paper describes the experiments conducted using HNC's CONVECTIS Indexing System for the TREC-4 routing and filtering tasks. An overview of CONVECTIS is described as well as the modifications to the system to perform both the routing and the filtering tasks. In prior participation in TREC the *MatchPlus* Context Vector Retrieval System [1] was utilized. The MatchPlus system relied on Context Vectors [3] as well as a broad Boolean filter for optimal performance. The Boolean filter created a large subset of likely relevant documents which were then ranked using the Context Vector technique. For TREC-4, as an experiment, the Boolean filter was not used. Instead, CONVECTIS was used which relied only on judged documents as prototypes of desired behavior. Preliminary results of these experiments are presented.

I. CONVECTIS INDEXING APPROACH

A. Overview

While the main function of CONVECTIS is to assign index terms to documents, CONVECTIS can be easily extended to handle traditional retrieval and routing. The following discussion refers to the assignment of an index term to documents. If by assignment of an index term the document is routed, CONVECTIS can also function as a document router. This is the manner in which CONVECTIS was used for the experiments conducted for TREC-4.

The approach to the indexing task is based upon the attributes of Context Vectors (for an overview of Context Vectors see [3]) and a key aspect of the indexing problem:

- Words that are used in a similar context convey similar meanings and will have context vectors that point in similar directions.
- Documents that discuss similar themes will have Context Vectors that point in similar directions.
- Documents that discuss a similar theme will have similar index terms.

The *MatchPlus* learning algorithm will produce word vectors that meet the above criterion. Since document vectors are a linear combination of word vectors, document vectors will meet the above criterion as well. Each index term can be associated with a basic concept. However, these concepts can be complex and possibly disjoint in the concept space. For example, the index term *telecommunications* could be defined to include the concepts ISDN, ATM, cellular phones, land lines, satellite communication channels and the internet. A single context vector per index term could be insufficient to correctly characterize a complex concept. As such, CONVECTIS is capable of using sets of context vectors to characterize index term concepts. This approach has a number of advantages:

- **Adjustable to Complexity of Index Concepts.** For those concept areas that are "simple", only a few, perhaps only one, vector will be required to characterize the concept. The number of vectors per index concept will be variable from index term to index term. The number of vectors required will be determined at system build time. If indexing performance for a particular index term becomes unacceptable, the number of vectors for that term can be increased to improve performance.
- **Easily Obtained Initial Concepts.** Initial conditions for the set of vectors for an index concept are easily obtained by using the document context vectors of documents that would be assigned to that index term. Although this step does require human intervention, it is easily performed by clerical-level personnel. One document must be chosen for each vector in an index term set. These vectors provide the initial condition for the tuning and refinement step.
- **Tunable Based on Incorporation of Human Feedback.** In addition to a set of vectors for each index term, a learning algorithm can be applied to allow incorporation of human

judgments as to the correctness of index terms being assigned by CONVECTIS. This approach offers the ability for human participation in the training process to the degree desired or required.

Given a set of documents and a predefined set of index terms, the indexing task is to assign to each document all index terms that apply. CONVECTIS provides facilities for a user to define index terms, create initial context vector representations for each term, and run the indexing process. CONVECTIS allows the user to inspect its assignments of index terms to documents, and make their own judgments about which index terms truly apply. Once user judgments are available, CONVECTIS can compute performance measures such as precision and recall. In addition, CONVECTIS can apply an automated learning algorithm to the human judgments in a form of relevance feedback to tune the index term context vectors for improved indexing performance.

An index term represents an *index concept*, which may be thought of as a region of the context vector meaning space. An index concept can be arbitrarily complex and may include disjoint portions of the context vector space. As a consequence, CONVECTIS assumes that a *set* of context vectors may be required to characterize each index concept. Indeed, regions of arbitrary complexity can be defined by using sets of vectors with adjustable vector dot product thresholds. A document will be assigned an index term if its context vector lies in the region defined by the index concept.

For simple concepts, only a few, perhaps only one, vector will be required to characterize the concept. The number of vectors per index concept may vary from index term to index term. If indexing performance for a particular index term is unacceptable, the number of vectors for that term can be increased to improve performance. To index a document, the context vectors for each index term are compared to the context vector for the document. The details are open to variation, but the algorithm currently used is quite simple and is shown in Equation 1.

$$\boxed{Score_{ij} = \max(D_i \cdot E_{ij}), i = 1, N; j = 1, M}$$

$Score_{ij}$ = the score of the current document D_i to index term j.

D_i = Document context vector of document to be indexed.

E_{ij} = Exemplar context vector k for index term j.

$\max()$ = the maximum operator

Equation 1. Assignment Decision

The dot products of the document context vector with each context vector for the index term are taken. If the maximum dot product is greater than a global threshold value, the index term is assigned to the document; otherwise, it is not assigned. CONVECTIS also supports the use of Boolean terms to modify the context vector assignments, as described further below.

This approach has a number of advantages. The key advantage is the ability to characterize a "diffuse" concept by the concept vector set and an optional set of Boolean inclusion/exclusion terms. Since the concept is represented by a set of vectors that are a purely mathematical representation, a learning algorithm can be used to "train" the vectors set based on human assessment of indexing performance. Yet another advantage is that the threshold of closeness is tunable and can be adjusted to insure maximum performance.

B. Index Term Initialization

The user must select the set of index terms. These concepts may already exist within the user's organization or may be identified as a result of analysis of document clusters. For each index term, a set of initial context vectors must be defined.

The effort required for initialization may be varied depending on the degree to which human judgments will be available to drive the automated learning process. If no human judgments will be available, for best performance the initial conditions should be determined with considerable care. However, if human judgments will be plentiful, the initial conditions are nearly irrelevant, as the learning algorithm is powerful enough to overcome poorly chosen ones. For greatest performance, it has proven most effective to rely on human judgments and the CONVECTIS learning algorithm rather than just initial conditions.

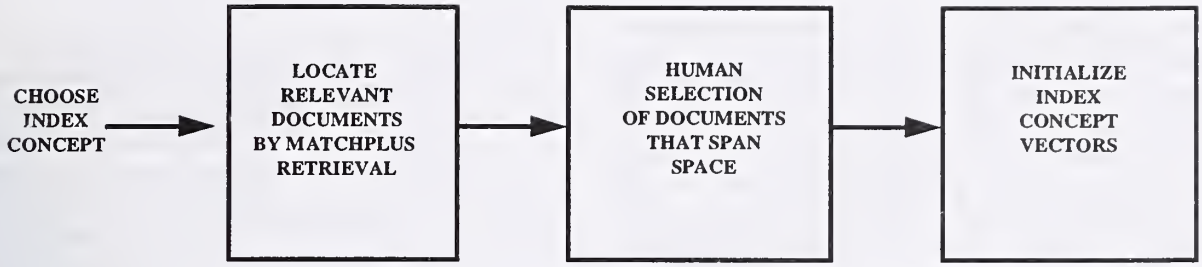

CHOOSE INDEX CONCEPT
LOCATE RELEVANT DOCUMENTS BY MATCHPLUS RETRIEVAL
HUMAN SELECTION OF DOCUMENTS THAT SPAN SPACE
INITIALIZE INDEX CONCEPT VECTORS

Figure 1. Index Concept Initialization.

Good initial conditions for the set of vectors for an index term are easily obtained by using the context vectors of documents that would be assigned that index term. Documents that an index term applies to may be located using any document retrieval method. If learning is not to be used, a person with domain expertise in the current concept should ensure that the set of documents selected covers the breadth of the concept space. If learning is to be used, a short free text description of the concept is quite adequate as an initial condition, and document retrieval capability is not necessary.

C. *Index Term Tuning*

Depending on the context vectors initially chosen to represent index terms, initial indexing performance may or may not be satisfactory. CONVECTIS incorporates an automated learning feature, based on relevance feedback from a human judge, that can substantially improve indexing performance.

CONVECTIS provides a simple but effective interface for obtaining user judgments. For each index term, the documents to which CONVECTIS assigned the index term are listed. The user simply reads the text of a document and decides whether the index term applies. If it does, he presses a button labeled "Applies", otherwise a button "Doesn't Apply". The text of the next document is automatically displayed. Making judgments is easily performed by clerical-level personnel once they understand the meaning of the index terms being used.

For any document and any index term, CONVECTIS either assigns the index term to the document or it does not. The human user may judge either that the index term does apply to the document or that it does not. CONVECTIS and the human may agree or disagree. If they disagree, the human judgment is assumed to be correct, and CONVECTIS' indexing process must be modified. In particular, the context vector representation of the index term must be adjusted.

Suppose the human determined that an index term applies to the document but CONVECTIS did not assign the term. This decision was reached because the maximum dot product of the exemplar set was not sufficient to warrant assignment of the index term. Since decisions are made on the basis of dot product, the exemplar vector must be moved *closer* to the

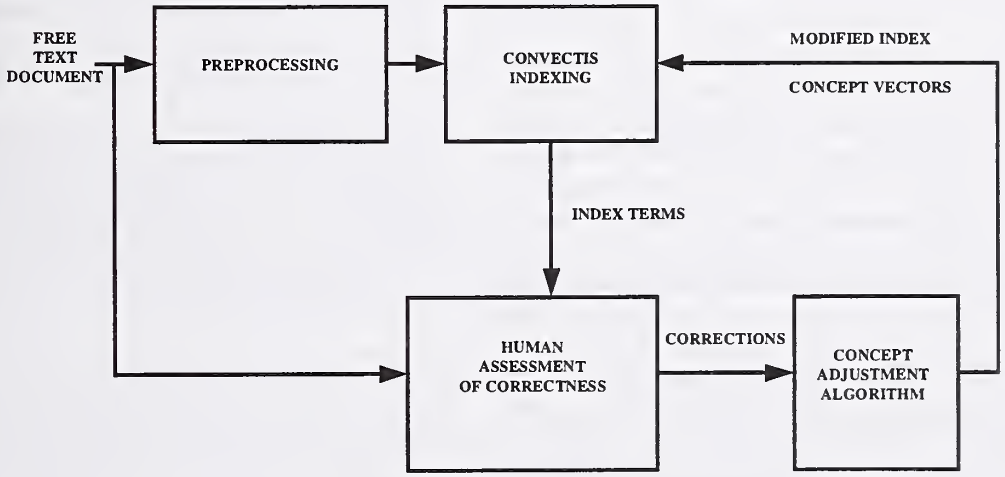


Figure 2. Tuning and Maintenance of Index Concepts.

current document vector to increase the dot product and result in the assignment of the term. If the human judged a term not to apply, but CONVECTIS assigned the term, the exemplar vector responsible for the decision must be moved *away* from the current document vector to reduce the strength of the association. This algorithm is closely related to the Learning Vector Quantization (LVQ) algorithm proposed by Kohonen [5]. The update algorithm is given in Equation 2.

$$\boxed{V_{ij}^{new} = V_{ij}^{old} + C_{ij}\left\|D_i - V_{ij}\right\| \cdot \gamma \cdot (D_i - V_{ij}^{old})}$$

V_{ij}^{new} = New CV "i" for index term "j" after update

V_{ij}^{old} = New CV i for index term j before update

D_i = CV of Document to be indexed

γ = learning rate, $0<=\gamma<=1$

C_{ij} = Correctness for index term j for document I

possible values for C_{ij} are as follows:

C_{ij} = 0, if CONVECTIS and human agree

C_{ij} = 1, if human assigned but CONVECTIS did not

C_{ij} = -1, if CONVECTIS assigned but human did not assign

Equation 2. Index Term Update

II. Approach to TREC-4

A. Modifications to CONVECTIS system

While the CONVECTIS system is designed to assign index terms, it was relatively straightforward to conduct both routing and filtering experiments. The "index terms" used were the 50 topics chosen for the routing and filtering tasks. CONVECTIS simply decided whether the index term (topic query) should be assigned to the documents (routed). Using the update equation (Figure 3) the set of relevance judgments were used to train the system. Indexing was run after each iteration and the performance was calculated. The vectors were again updated according to the update equation and the performance was characterized. This process was repeated until the optimal performance was achieved.

B. Experiments

Approximately 2/3 of the relevance judgments were used for training. The update equation was applied, then routing was performed on the training and the test set. This process was continued until the optimal performance (in terms of precision and recall) was obtained on the test set. These tests were designed to test the effectiveness of the update equation both in terms of performance and stability.

As described in the CONVECTIS overview it is possible to have more than one Context Vector per topic query. Experiments using one, three and ten Context Vectors were conducted.

C. Results

In order to determine the effectiveness and stability of the learning algorithm and to determine the number of learning iterations necessary for optimal performance the TREC scoring algorithm was used after each learning iteration. The relevant retrieved and the average precision are presented for a run which used 1 context vector per topic query in Table 1.

Learning Iteration	Relevant Retrieved	Average Precision
1	7235	.1535
2	9872	.2240
20	11477	.2990
30	11860	.3232
50	12226	.3464
100	12571	.4227
200	12586	.4221

Table 1. Learning Iterations

Clearly the performance of the system improves as a function of the number of iterations. Also important is the convergence of the learning algorithm as evidenced by the very slight change in performance between 100 iterations and 200 iterations.

The experiments involving multiple Context Vectors for each topic query are presented in Table 2. Unlike the case with just one Context Vector per topic query the initial vector cannot be set to zero. A randomly chosen relevant document was used for the initial vector before training took place.

Number of Vectors	Relevant Retrieved	Average Precision
1	7502	.2492
3	7048	.2275
10	6791	.1802

Table 2. Multiple Context Vectors (Testing)

This is a somewhat surprising result. It was thought that for complex concepts such as *telecommunications* (as described in Section I.A) a single Context Vector would not be sufficient to characterize all its meanings. In an effort to gain a better understanding of how the multiple Context Vector routing queries are behaving statistics about the number of documents that were routed because of each vector were tracked. It was thought that a single vector might be responsible for all the routed documents or that one vector was causing all the routing of non-relevant documents. Table 3 presents some of those statistics for a system with 3 context vectors per topic query. The number of documents that were acted upon for each context vector is presented.

Topic	Context Vector	a	b	c	d
20	1	33	20	1	56
	2	56	41	4	97
	3	48	35	6	92
40	1	56	22	0	42
	2	35	20	0	42
	3	52	0	0	22
143	1	114	28	24	149
	2	169	64	50	156
	3	55	5	8	65
191	1	25	0	0	53
	2	36	15	20	154
	3	34	1	21	143

a: Number of documents routed and relevant
b: Number of documents not routed and relevant
c: Number of documents routed and not relevant
d: Number of documents not routed and not relevant

Table 3. Multiple CV Distribution

Table 3 shows that for each of the topics no single context vector is responsible for the decisions. When multiple context cectors are used to represent each topic query each Context Vector is being used in creating the routing decision surface.

Additional experiments need to be conducted to better understand why the single Context Vector outperforms the systems using multiple Context Vectors. This includes investigation of the training algorithm for multiple Context Vectors, adjusting the decision thresholds even perhaps on a per topic basis and the selection method for the initial conditions of the vectors prior to training.

D. Discussion

The most likey reason for the poor performance of the system using multiple context vectors for each topic query is over parameterization. Evidence of this fact can be seen in Table 4. This table presents the performace of the system using 1, 3 and 10 context vectors on the training set (as opposed to Table 2 which is on the test set).

Number of Vectors	Relevant Retrieved	Average Precision
1	4095	.5320
3	4940	.7189
10	5591	.8424

Table 4. Multiple Context Vectors (Training)

Clearly the performance of the system on the training set improves as more vectors are used. With multiple vectors CONVECTIS has learned the training set very well but has not craeted a general vector decision space that will work well with documents outside the test set.

III. Conclusion

By participating in TREC-4, HNC gained very useful insight into the behavior of the CONVECTIS system. While the results were not as good as our previous participation in the TREC conferences [1,2], again most likely due to the missing Boolean filter, the experiments conducted have suggested many potential enhancements and improvements to the CONVECTIS indexing system.

This was the first large scale test of CONVECTIS. The use of a large number of relevance judgements (with a high precentage of non-relevant judgements versus relevant judgements) had never been tried.

IV. References

[1] S.I. Gallant, W.R. Caid, et al, "HNC's *MatchPlus* System", in *Proceedings TREC-1 Conference*, D. Harman, ed, NIST Special Publication 500-207, Rockville, MD. Nov. 1992.

[2] S.I. Gallant, W.R. Caid, et al, "Feedback and Mixing Experiments with *MatchPlus*", in *Proceedings TREC-2 Conference*, D. Harman, ed, NIST Special Publication 500-215, Rockville, MD. Nov. 1993.

[3] W.R. Caid and J.L. Carleton, "Context Vector-Based Text Retrieval", in *Proceedings IEEE Dual-Use Conference*, 1994.

[4] R.V. Sasseen and W.R. Caid, "Docuverse: A Context Vector-Based Approach to Graphical Representation of Information Content", in *Proceedings IEEE Dual-Use Conference, 1994.*

[5] T. Kohonen, *Self-Organization and Associative Memory*, 2nd ed., Springer-Verlag, 1988.

Document Routing by Discriminant Projection : TREC-4

Kok F. Lai [1] Vincent A.S. Lee Jeremy P. Chew [2]

Information Technology Institute
11 Science Park Road, Singapore 0511.

Abstract

We present document routing as a standard problem in discriminant analysis. The standard solution involves the inversion of a large matrix whose dimension is the number of indexed terms. Typically, the solution does not exist because the number of training documents are much smaller compared to the number of terms. We show that one can project this raw document space into a lower dimensional space where solution is possible. Our projection algorithm exploits the characterisitics of the empty space, using only the training documents for efficient coding of the relevance information. Its complexity is linear with respect to the number of terms, and second order with respect to the number of training documents. We can therefore fully exploit the power of discriminant analysis without imposing severe computational and storage constraints.

1 Introduction

This paper describes the experiments performed by the Information Technology Institute for TREC-4. We are participating in TREC for the first time and have elected to perform only the main routing experiment.

We adopt a vector space model and attempt to construct the optimum routing queries using only the training documents. The subsequent sections contain the details of our *discriminant projection algorithm* which was used for this purpose.

We present the routing task as a standard problem in discriminant analysis. The standard solution involves the inversion of a large matrix whose dimension is the number of indexed terms. However, it is usually not possible to invert this matrix because the number of training documents are much smaller compared to the number of terms. This phenonenum is commonly called the *empty space* syndrome.

Consequently, it is desirable to project the raw document space into a lower dimensional space where each axis possesses higher representational richness. We show that linear orthonormal projections that efficiently encode the relevance information can be constructed using training documents only. We describe a projection algorithm that exploits the peculiarity of the empty space to construct the orthogonal basis on-the-fly. The complexity of this algorithm is linear with respect to the number

[1] Please direct all enquiries and correspondence to Kok F. Lai, Information Technology Institute, 11 Science Park Road, Singapore. Email address : kflai@iti.gov.sg.

[2] On attachment from Nanyang Technological University, Singapore, from Jan 8, 95 to Jun 18, 95.

of terms and second order with respect to the number of training documents. It first constructs a series of orthogonal axis using the Gram-Schmidt procedure, and then rotates the reduced space so that the axis are ordered by their importance. One can then discard insignificant axis to obtain an efficient representation of relevance information.

We then describe how one can perform discriminant analysis in this reduced space, and obtain the routing queries by projecting the results back into the original space. As we construct the projection basis on-the-fly, it is not necessary to alter the indexing structures, and no additional storage is required.

Due to time and resource contraints, we made several approximations in our TREC-4 implementations and these are described in this report. We will attempt to lift these approximations to improve the system's performance in subsequent participations.

We begin in the next section by describing the discriminant projection algorithm.

2 The Discriminant Projection Algorithm

2.1 Construction of Routing Query using Discriminant Analysis

We denote a document as a vector as follows :

$$\mathbf{d} = [f_1, f_2, \dots f_N]^T \tag{1}$$

where f_j is a function of the term frequency for term j. N is the total number of unique indexed terms, and is typically very large. We shall assume that $\mathbf{d}$ is normalized, *i.e.*, $\|\mathbf{d}\| = 1$, in subsequent discussions.

Denote a document collection as $\mathbf{D} = [\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_M]$. For a typical routing task, $\mathbf{D}$ consists of three separate groups : $\mathbf{D}_r$, $\mathbf{D}_{nr}$, and $\mathbf{D}_{na}$. $\mathbf{D}_r$ contains the set of M_r relevant documents; $\mathbf{D}_{nr}$ contains the set of M_{nr} non-relevant documents; and $\mathbf{D}_{na}$ contains the set of M_{na} documents where relevance information is unavailable.

Note that the total number of document, $M = M_r + M_{nr} + M_{na}$. Without loss of generality, let

$$\mathbf{D} = [\mathbf{D}_r : \mathbf{D}_{nr} : \mathbf{D}_{na}] \tag{2}$$

We want to find a *routing query* $\mathbf{q}$, such that $\mathbf{D}_r^T\mathbf{q} = \mathbf{1}$ and $\mathbf{D}_{nr}^T\mathbf{q} = \mathbf{0}$. Thus, we write

$$[\mathbf{D}_r : \mathbf{D}_{nr}]^T\mathbf{q} = [\mathbf{1} : \mathbf{0}]^T \tag{3}$$

Pre-multiplying both sides of the equation by $[\mathbf{D}_r : \mathbf{D}_{nr}]$, we have

$$\mathbf{R}\mathbf{q} = \mathbf{D}_r\mathbf{1} \tag{4}$$

where

$$\mathbf{R} = [\mathbf{D}_r : \mathbf{D}_{nr}]\begin{bmatrix} \mathbf{D}_r^T \\ \mathbf{D}_{nr}^T \end{bmatrix} \tag{5}$$

Now, observe that

$$\mathbf{D}_r\mathbf{1} = \sum_{i=1}^{M_r} \mathbf{d}_{r_i} = M_r\mathbf{m} \tag{6}$$

where $\mathbf{m}$ is the *mean* or *centroid* of the relevant documents. Suppose that the NxN matrix $\mathbf{R}$ is non-singular, we may then solve for $\mathbf{q}$ as follows:

$$\mathbf{q} = M_r \mathbf{R}^{-1} \mathbf{m} \tag{7}$$

Note that equation (7) is equivalent to the well-established *discriminant analysis* in statistical pattern recognition.

However, in most routing applications, the number of terms, N, is very much larger than the number of training documents $M_r + M_{nr}$. Thus $N >> M_r + M_{nr}$ and $\mathbf{R}^{-1}$ does not exist. Consequently it is impossible to solve for $\mathbf{q}$ from equation (7).

2.2 Solving Discriminant Analysis in a Reduced Space

Recall from the previous section that $N >> M_r + M_{nr}$ and $\mathbf{R}$ is singular. To solve for $\mathbf{q}$, it is necessary to project the raw document space into a lower dimensional space where each axis possesses higher representational richness.

We can project the raw document space $\mathbf{D}$ into a lower dimensional space $\mathbf{V}$ as follows:

$$\mathbf{V} = \mathbf{U}^T[\mathbf{D}_r : \mathbf{D}_{nr}] \tag{8}$$

where $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_k]$ are the first k orthonormal basis that account for most information in the raw space. Using $\mathbf{V}$, we can then obtain the transformed routing query $\mathbf{q}_v$ as follows:

$$\mathbf{q}_v = M_r \mathbf{\Lambda}^{-1} \mathbf{m}_v \tag{9}$$

where $\mathbf{m}_v$ is the mean vector of relevant documents in the transformed space. Now

$$\mathbf{\Lambda} = \mathbf{V}\mathbf{V}^T = diag(\lambda_1, \lambda_2, \ldots, \lambda_k) \tag{10}$$

is a diagonal matrix, where λ_i are the eigenvalues of $\mathbf{R}$ in equation (5) in descending order. Note that $\mathbf{\Lambda}^{-1} = diag(\lambda_1^{-1}, \lambda_2^{-1}, \ldots, \lambda_k^{-1})$ exists and thus $\mathbf{q}_v$ can be solved via equation (9).

We can then obtain the original query $\mathbf{q}$ in the raw space via

$$\mathbf{q} = \mathbf{U}\mathbf{q}_v = \sum_{i=1}^{k} q_{vi}\mathbf{u}_i \tag{11}$$

The above formulation is related to singular value decomposition[3] (SVD) employed by Latent Semantic Indexing [2]. However, the order of complexity of the standard SVD algorithm [3] is $O(N^2 + k^3)$. Since the total number of terms N is very large ($> 10^6$) for a typical application, this is very computational intensive. Moreover, SVD is usually pre-computed based on the entire document collection, which includes documents that carry no relevance information. As a result, the reduced dimensional space does not efficiently represent the relevance information contained in the training documents. The pre-computed SVD projection also substantially increases storage requirements [4].

We present an algorithm below, called *semantic projection*, that uses only the training documents to obtain the orthonormal basis. The algorithm reduces the order of computation substantially to approximately $O(N(M_r + M_{nr})^2)$. This allows it to compute the orthonormal basis on-the-fly and thus no additional storage requirement is necessary.

[3] Please refer to [1] for the relationships between singular values and eigenvalues.

2.3 Semantic Projection

Recall from the previous section that we wish to find the orthonormal basis $\mathbf{U}$ such that

$$\mathbf{V} = \mathbf{U}^T[\mathbf{D}_r : \mathbf{D}_{nr}] \tag{12}$$

It can be shown [1] that $\mathbf{U}$ can be computed via eigen-decomposition of the NxN matrix $\mathbf{R} = [\mathbf{D}_r : \mathbf{D}_{nr}][\mathbf{D}_r : \mathbf{D}_{nr}]^T$. However, due to the peculiar nature of text routing applications where frequently $N >> M_r + M_{nr}$, it will be advantageous to find an alternative method to compute $\mathbf{U}$.

The *semantic projection* algorithm uses only the $M_r + M_{nr}$ documents with relevance information to compute the optimum projection. It first contructs a subspace based on the training documents using the Gram-Schmidth Procedure. The dimension of this subspace will be at most $M_r + M_{nr}$. It then performs eigen-decomposition on this subspace to extract orthonormal basis which are ordered by their importance in capturing the subspace information. By specifying the desirable representation richness, only $k < M_r + M_{nr}$ orthonormal basis needs to be extracted.

2.3.1 Gram-Schmidt Procedure

The procedure makes use of training documents $\mathbf{d} \in [\mathbf{D}_r : \mathbf{D}_{nr}]$. We begin the procedure with the initialization

$$\mathbf{y}_1 = \mathbf{d}_1 \tag{13}$$

We then construct $\mathbf{y}_2$ as follows :

$$\begin{aligned} \mathbf{x}_2 &= \mathbf{d}_2 - \mathbf{d}_2^T \mathbf{y}_1 \mathbf{y}_1 \\ \mathbf{y}_2 &= \frac{\mathbf{x}_2}{\|\mathbf{x}_2\|} \end{aligned} \tag{14}$$

The vector $\mathbf{y}_2 = \mathbf{0}$ if and only if $\mathbf{d}_2 = a\mathbf{d}_1$, meaning that $\mathbf{d}_2$ is linearly dependent upon $\mathbf{d}_1$. Assume that $\mathbf{y}_2 \neq \mathbf{0}$, then $\mathbf{y}_1$ and $\mathbf{y}_2$ are linearly independent and orthogonal :

$$\mathbf{y}_2^T \mathbf{y}_1 = \mathbf{0} \tag{15}$$

We continue this procedure, generating each new vector $\mathbf{y}_j$ as follows :

$$\begin{aligned} \mathbf{x}_j &= \mathbf{d}_j - \sum_{i=1}^{j-1} (\mathbf{d}_j^T \mathbf{y}_i)\mathbf{y}_i \\ \mathbf{y}_j &= \frac{\mathbf{x}_j}{\|\mathbf{x}_j\|} \end{aligned} \tag{16}$$

This procedure maps the vectors $(\mathbf{d}_1, \mathbf{d}_2, \ldots \mathbf{d}_{M_r+M_{nr}})$ into vectors $(\mathbf{y}_1, \mathbf{y}_2, \ldots \mathbf{y}_{M_r+M_{nr}})$. In the latter set, there will be $r \leq M_r + M_{nr}$ nonzero vectors that the linearly independent which are retained. The discarded vectors are linearly dependent on the r linearly independent vectors and do not carry additional information on the training documents.

Note that the complexity of this algorithm is $O(Nr^2)$ and is linear with respect to the number of terms N. In the worst case, the complexity will be $O(N(M_r + M_{nr})^2)$.

2.3.2 Eigen-Decomposition of the Resultant Subspace

Without loss of generality, we write the resultant nonzero vectors as follows : $(\mathbf{y}_1, \mathbf{y}_2, \ldots \mathbf{y}_r)$. We can obtain the rxr correlation matrix $\mathbf{R}_y$ in this space by simply computing the projection on each training documents on the $\mathbf{y}$ vectors :

$$\mathbf{R}_y(i,j) = \sum_{m=1}^{r} \mathbf{d}_m^T \mathbf{y}_i \cdot \mathbf{d}_m^T \mathbf{y}_j \tag{17}$$

We can then apply an eigen-decomposition algorithm to efficiently extract eigenvectors of this r-dimensional subspace. In our implementation, we use the Hotelling's Iterative Procedure [5] to solve the eigen-decomposition problem. The procedure iteratively obtain eigenvectors $\mathbf{w}_1, \mathbf{w}_2, \ldots$ that corresponds to eigenvalues $\lambda_1, \lambda_2, \ldots$ in decending order. One can simply terminate the procedure if a pre-specified threshold t has been exceeded :

$$\sum_{i=1}^{k} \lambda_i / tr(\mathbf{R}_y) \geq t \tag{18}$$

$tr(\mathbf{R}_y)$ is the trace of the matrix $\mathbf{R}_y$ that denotes the *total* variance available in the training documents. For example, if $t = 0.75$, then the first k eigenvectors account for more than 75% of the information in the training documents.

In the Hotelling's procedure, the iteration of each eigenvectors typically take only a few steps to converge. Since the algorithm operates on $\mathbf{R}_y$, its complexity is $O((M_r + M_{nr})^2)$ in the worst case.

Finally, the transformation vectors $\mathbf{u}$ can be obtained via

$$\mathbf{u}_i = \sum_{j=1}^{k} w_{ij} \mathbf{y}_j \tag{19}$$

Each $\mathbf{u}_i$, in diminishing importance, captures an orthogonal axis containing the semantics of the training documents. Thus $\mathbf{u}_1$ is a linear combination of terms containing the highest information contents in the training documents. $\mathbf{u}_1$ is orthogonal to $\mathbf{u}_2$ *i.e.*, $\mathbf{u}_1^T \mathbf{u}_2 = 0$: it captures information contents missed by $\mathbf{u}_1$, and so on. This set of orthonormal basis projects the original raw space into a reduced dimensional space while retaining the semantic information :

$$\mathbf{V} = \mathbf{U}^T [\mathbf{D}_r : \mathbf{D}_{nr}] \tag{20}$$

Thus the reduced space $\mathbf{V}$ is called the *semantic projection* of the training documents.

2.4 The Discriminant Projection Algorithm : Summary

In summary, the proposed routing algorithm is based on the well-established *discriminant analysis* as shown in equation (7). However, equation (7) by itself is unsolvable due to the *empty space* syndrome. The algorithm projects the training documents into a reduced space using a method called *semantic projection*, and solves for the routing query in this reduced space. The resulting query vector is then projected back into the original space.

The complexity of this algorithm is linear with respect to the number of terms N, and square with respect to the number of training documents $M_r + M_{nr}$. This makes it particularly attractive in relevance feedback environment where the number of feedback documents is typically very small.

We list the steps of the proposed *discriminant projection algorithm* below. Note that with a little housekeeping, steps 2 and 3 can be combined into a single operation.

1. Collect training documents $[\mathbf{D}_r : \mathbf{D}_{nr}]$. Obtain the mean vector $\mathbf{m}$ of the relevant documents by $\mathbf{m} = \frac{1}{M_r}\sum_{i=1}^{M_r} \mathbf{d}_{r_i}$.
2. Perform Gram-Schmidt procedure to obtain the r linearly independent vectors.
3. Obtain the rxr matrix $\mathbf{R}_y$ by computing the projection of each training documents with the r linearly independent vectors.
4. Use Hotelling's procedure to extract eigenvectors until required threshold is reached. Put the resulting k eigenvectors $(\mathbf{u})$ into the projection matrix $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \ldots \mathbf{u}_k]$, and the eigenvalues (λ) into the diagonal matrix $\mathbf{\Lambda} = diag(\lambda_1, \lambda_2, \ldots, \lambda_k)$.
5. Project $\mathbf{m}$ to the reduced space : $\mathbf{m}_v = \mathbf{U}^T\mathbf{m}$.
6. Obtain the transformed routing query $\mathbf{q}_v = M_r\mathbf{\Lambda}^{-1}\mathbf{m}_v$.
7. Finally, obtain the routing query $\mathbf{q}$ in the original space by $\mathbf{q} = \mathbf{U}\mathbf{q}_v = \sum_{i=1}^{k} q_{vi}\mathbf{u}_i$.

3 TREC IV Implementations

As a first-time participant, we decided to take part in the main routing task only. Due to time and resource constraints, we experimented only with documents that appear on the relevant list of at least one query. All other documents that do not belong to this group are simply ignored. Consequently our training documents are drawn only from approximately 18,000 documents, rather than from the entire TREC collection.

Each document is automatically parsed and index terms representing the document content are retrieved. A simple token recognizer written in flex is used to identify words within the document structure that contributes to the content. Most words are indexed except very common words, maintained in a stopword list, and numbers are removed. The remaining words are then stemmed using the Porter's stemming algorithm. The document indexing process next creates the inverted files using Faircom's c-tree Plus file management product. Currently, without compression, the overhead for the indices amounts to about 80% of the original database size.

Using the training documents, we compute the *idf* weights for each term, *i.e.* $idf_i = log(N/n_i)$. Weights of terms seen only in test documents but not in training documents are set to 1.

For our first experiment, *itidp1*, we made no use of the topics and used only the relevant documents to construct the queries $(M_{nr} = 0)$. Using this special case, it can be shown that step 5 of the algorithm reduces to

$$\mathbf{m}_v = \mathbf{U}^T\mathbf{m} = [1, 0, 0, \ldots, 0]^T \tag{21}$$

and therefore $\mathbf{q} = \mathbf{m}$. Thus the optimum routing query is simply the mean (centroid) of the relevant documents.

For our second experiment, *itidp2*, we enriched the training documents in *itidp1* by drawing from the 18,000 documents not marked relevant for the particular query. To further reduce the computational complexity, we did not invoke the Gram-Schmidt procedure for the non-relevant documents, and used these only to compute the projection into $\mathbf{R}_y$. This caused the complexity of the training algorithm to remain as $O(NM_r^2)$, in the expense of reduced representational richness.

For both sets of queries, we compute the projection scores $\mathbf{q}^T\mathbf{d}$ for the test documents $\mathbf{d}$, and ranked the result list in decending order.

4 Results

The following are the results of our implementation :

```
Queryid (Num):       all  itidp1
Total number of documents over all queries
    Retrieved:    42000
    Relevant:      5091
    Rel_ret:       3537
Interpolated Recall - Precision Averages:
    at 0.00       0.6882
    at 0.10       0.4499
    at 0.20       0.3763
    at 0.30       0.3209
    at 0.40       0.2641
    at 0.50       0.2257
    at 0.60       0.1821
    at 0.70       0.1304
    at 0.80       0.0660
    at 0.90       0.0234
    at 1.00       0.0002
Average precision (non-interpolated) over all rel docs
                  0.2282
Precision:
  At    5 docs:   0.5000
  At   10 docs:   0.4714
  At   15 docs:   0.4206
  At   20 docs:   0.4071
  At   30 docs:   0.3794
  At  100 docs:   0.2919
  At  200 docs:   0.2270
  At  500 docs:   0.1388
  At 1000 docs:   0.0842
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.2782


Queryid (Num):       all  itidp2
Total number of documents over all queries
    Retrieved:    42000
    Relevant:      5091
    Rel_ret:       3104
```

```
Interpolated Recall - Precision Averages:
    at 0.00       0.6528
    at 0.10       0.4252
    at 0.20       0.3123
    at 0.30       0.2510
    at 0.40       0.1961
    at 0.50       0.1457
    at 0.60       0.1041
    at 0.70       0.0604
    at 0.80       0.0249
    at 0.90       0.0096
    at 1.00       0.0007
Average precision (non-interpolated) over all rel docs
                  0.1768
Precision:
  At    5 docs:   0.4714
  At   10 docs:   0.4524
  At   15 docs:   0.4254
  At   20 docs:   0.4083
  At   30 docs:   0.3738
  At  100 docs:   0.2664
  At  200 docs:   0.1973
  At  500 docs:   0.1177
  At 1000 docs:   0.0739
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.2221
```

5 Conclusions

We presented a method for document routing using the discriminant projection algorithm. As a first-time participant in TREC, we faced time and resource contraints and had to make several approximations in our implementation. We wish to lift these approximations in our subsequent participation in TREC.

References

[1] L. L. Scharf, *Statistical Signal Processing: Detection, Estimation and Time Series Analysis*, Addison-Wesley, 1991.

[2] S. T. Dumais, G. W. Furnas & T. K. Landauer, "Indexing by Latent Semantic Analysis," *Journal of the American Society for Information Science*, 1990, pp. 391-407.

[3] R. A. Harshman & M. E. Lundy, "Data preprocessing and the extended PARAFAC model," *Research Methods for MultiMode Data Analysis*, edited by H. G. Law et. al., 1984.

[4] D. Hull, "Improving Text Retrieval for the Routing Problem using Latent Semantic Indexing," *Proc SIGIR '94*, 1994, pp. 282-291.

[5] H. Hotelling, “Analysis of a Complex Statistical Variables into Principal Components,” *Journal of Educational Psychology*, 1933, pp. 417-441, 498-520.

Boolean System Revisited: Its Performance and its Behavior

X. Allan Lu*, John D. Holt, David J. Miller

Lexis-Nexis, a division of Reed Elsevier Inc.
9555 Springboro Pike
Miamisburg, Ohio 45342

Telephone: 513-865-6800 ext. 5404
Fax: 513-865-1655
alan, johnh,davidm@lexis-nexis.com

ABSTRACT

This experimental study attempts to provide a general conclusion to the Boolean information retrieval system regarding its performance on retrieval effectiveness and its behavior on retrieval of different relevant documents. Specifically a representative commercial Boolean system is compared to the best ranking systems reported in TREC4. The Boolean system delivered a comparable performance and retrieved a set of rather unique relevant documents. The study also justifies its methodology for comparing non-ranking and ranking systems.

1.0 Introduction

Boolean search has been dominating the commercial information retrieval for over 30 years and a large number of professional searchers have mastered the Boolean search tool. The size of the Boolean searcher community forces the latest information retrieval software vendors to provide this search feature. But the Boolean search is an aged search technique. The criticism to the Boolean search system range from its requirement of intensive user training to its awkwardness in expressing complex conceptual relationships in queries, and to the absence of relevance ranking. The newly commercialized natural language search systems, like Freestyle™, are free from these problems and enable the professionals as well as the end-users to access the large commercial full-text databases.

An interesting question arises: should the Boolean search system be replaced entirely by the natural language search system that has relevance ranking capability? A positive answer to this question requires that the natural language search system is better than the Boolean system in terms of retrieval effectiveness, and that the natural language search system is capable of finding a similar set of relevant documents that the Boolean system tends to find. The motivation for having a better performance is obvious, the motivation for retrieving a similar set of relevant documents is to demonstrate to the user the redundancy of the Boolean system.

The information retrieval literature in the past suggests that in terms of search query formulation the simple relevance ranking system that accepts natural language queries usually performs as good as the Boolean system that provides complex querying language [1,2,3,4,5,6,7]. However these studies used very small test databases and their results suffered the lack of generalizability. A more recent study [8] using much larger test database suggested that the relevance ranking system could perform better than the Boolean system. Unfortunately the study used only legal data instead of more general materials, and employed only one type of relevance ranking system. In addition the study had the methodological issue of comparing relevance ranking to chronological ordering that definitely favored the relevance ranking system. Subject relevance and subject currency are two different relevance factors. The Boolean searcher attempts to incorporate both factors in the search in order to read recent and relevant documents first. By contrast the ranking system can only handle the single factor--subject relevance, indicating that the traditional ranking principle is rather rigid and inflexible in the environment of man-machine interaction. As a result, the ranking system may maintain a better performance by ranking the old and relevant document higher than the recent and relevant while the Boolean system may suffer given the same ranking sequence. For a better experimental design it is prudent to retreat to only compare the density of retrieved relevant documents in the Boolean and ranking systems, leaving the issue of determining which relevance factor, subject relevance or subject currency, is more important to the searcher.

These reported studies also failed in looking into the retrieved documents to examine whether the retrieved from different systems are similar or significantly different regardless their recall and precision performances. When there are many relevant documents available the systems with comparable performance may have different retrieval behavior, that is, they retrieve different sets of the relevant documents. Specifically, the Boolean searcher would be reluctant to migrate to the natural language search system if the Boolean system tends to find different relevant documents. Thus it is desirable to look beyond the traditional effectiveness measures when comparing the Boolean system and the ranking system.

TREC4 [9] presents an excellent opportunity to conduct a thorough comparison of the Boolean system and the relevance ranking system. First of all, the test data is sufficiently large and diverse, putting adequate stress on the involved systems. Particularly neither the Boolean search subject nor the ranking system designer is able to guess the amount of relevant documents in the test databases. Second, the test queries are much shorter than those used in the first three TRECs, with the average size (i.e., number of searchable words) close to the query size in the commercial retrieval environment [10]. Without the short queries the experimental design will favor the relevance ranking system and punish the Boolean system. Third, the search instructions designed for the interactive search track in TREC4 prevent any Boolean search subject from creating an answer set that is either very large to boost recall to an artificially high level or extremely small to ensure unrealistically high precision. The instructions are recall as well as precision neutral. The search time of thirty minute also prevent any Boolean search subject from constructing very knowledgeable Boolean queries, making the Boolean search similar to those performed in the real situations. Fourth and finally, the good systems identified in the ad hoc tests represent a group of different relevance ranking systems, making more general observations feasible. Better yet, the ad hoc search with manual query expansion in TREC4 is very similar to the Boolean search from the search procedure point of view.

2.0 Experiment Design

The Boolean search results are the results that Lexis-Nexis submitted to the interactive search track in TREC4. Briefly, each of the 10 Boolean search subjects, the Boolean search experts from the customer services department, was given 5 of the 50 TREC4 ad hoc topics in the sequence from 201 to 250. The topic assignment ignores any specific subject expertise on the part of any given subject. The databases for the ad hoc tests in TREC3 and TREC4 were loaded into the Lexis-Nexis online system for the subjects to access them through a Boolean querying interface. The interface provides every Boolean search feature that one can find in the commercial online systems, including word truncation and proximity. Some subjects interacted with the TREC3 queries and data for getting familiar with the TREC environment. The subjects were instructed to complete each search topic within 30 minutes and to create an answer set that is fairly precise and comprehensive as they usually do for their customers. The test searches were carried out by the subjects, in their offices, on a time available basis. In most cases, the topics were begun and finished without interruption. However, in some cases, a topic was saved, while business demands had to be met. This environment accounts for some of the lengthy times associated with a few topics.

For the purpose of this study the Lexis-Nexis entry for the interactive track did a few tasks that were neither required nor instructed. For maintaining the sample size the Lexis-Nexis entry did all 50 topics instead of the required 25 for the interactive search task. Next, unlike other interactive search entries in TREC4 the final answer set for every test topic represents the complete document set from a single Boolean search without reviewing each individual retrieved document. That is, the subjects did not manually construct the final answer set to include those documents deemed relevant by them, or to exclude those documents deemed not relevant by them. This is the major reason why the sizes of the answer sets have a wider range (Table 1) than those from other interactive systems. This is also the reason that the Boolean search results are more comparable to those of the manual ad hoc in TREC4 where the designer of the ranking system manually built the final search queries.

Finally the number of interactions with the system is very limited and in some cases is zero. The limited number of interaction with the system enhance the comparability of the Boolean search results to those from the ranking systems.

Topic	Retrieved	Topic	Retrieved
201	87	226	17
202	103	227	59
203	19	228	223
204	111	229	42
205	34	230	14
206	26	231	88
207	42	232	33
208	46	233	34
209	76	234	16
210	47	235	99
211	150	236	16
212	63	237	181
213	31	238	65
214	10	239	40
215	18	240	288
216	32	241	17
217	19	242	61
218	5	243	21
219	21	244	77
220	18	245	23
221	247	246	6
222	56	247	53
223	148	248	28
224	47	249	20
225	127	250	164

TABLE 1. **Boolean Answer Set Sizes**

The Boolean search results can be summarized as follows: The range of search time is from 4 and half minutes to 143 minutes, the average search time is 38 minutes, and the median search time is 30 minutes. The average micro-precision and micro-recall [11] are 0.4 and 0.17, respectively, and the average macro-precision and macro-recall [9] are 0.38 and 0.22, respectively. The composed E measure with alpha=0.5 is 0.76.

The relevance ranking systems in this study are the four best performers in the automatic and manual ad hoc tests [6]. Since both the Boolean system and the manual ad hoc systems had human involvement, their results are more comparable. The automatic ad hoc systems were included and compared for drawing a general performance conclusion.

The four best automatic ranking systems are "CrnlAE" from Cornell University, "pircs1" from Queen's College, "cityal" from City University of London, and "INQ201" from University of Massachusetts. Later another system "CrnlAL" from Cornell University was added to this group. The four best manual systems are "CnQst2" from Excalibur Technologies Inc., "pircs2" from Queens College, "uwgcl1" from University of Waterloo, and "INQ202" from university of Massachusetts.. For each of the selected ranking systems its 50 rank lists submitted to NIST for scoring were used for comparison. The query relevance table was also used for extracting the relevant documents from the rank lists.

To make the Boolean search results comparable to the ranking results, this study turned every relevance rank list into a document set using the size of the corresponding Boolean answer set generated for the same query [5]. The major benefit of this method is its focus on the process of search, leaving out the process of presenting search results and the associated relevance factors such as subject relevance and subject currency, and allowing a more static comparison with a better experimental control. The other arguments in the introduction section regarding the favorable TREC4 environment further reduce the potential bias usually associated with this method.

The actual design can be easily represented in a form of a matrix with its rows as queries and its columns as various systems, Boolean or ranking. Each cell in the matrix has two items, precision ratio and recall ratio calculated for a particular system and a particular query. Since there are 49 (Topic 201 was removed from the test by NIST) queries and 10 systems, the matrix has 490 cells or 490 pairs of precision and recall ratios. The populated matrix invites a typical one-way ANOVA analysis with both precision and recall as dependent variables and system as independent variable. The results from the one-way ANOVA test should indicate whether there is any significant performance difference among the 10 systems. When such difference is detected, the further analysis like Tukey's Studentized Range (HSD) Test or pairwise T-test can be employed to find out the system(s) that is different from the rest. MANOVA test can also be applied to this matrix to obtain the multivariate statistics of the two dependent variables.

This study takes an additional step to investigate whether the different systems tend to retrieve different documents or more specifically different relevant documents in a quantitative term. For this task a matrix with the same dimensions as the precision-recall matrix described in the preceding paragraph is populated with the number of unique documents retrieved and with the number of unique relevant documents retrieved in each cell. Once the populated matrix is ready, the same one-way ANOVA test and the same MANOVA test can be applied to it, with both the number of unique documents and the number of unique relevant documents as dependent variables and system as independent variable. The test results should provide the insight as to whether the systems tend to retrieve a similar amount of unique documents and unique relevant documents.

A definition for unique documents and unique relevant documents is in order. Initially the unique document to a system was defined as a document that was only retrieved by that system. The same definition applied to the unique relevant documents. Since the Boolean system is more comparable to the manual ad hoc systems, this study splits the matrix into two, the first behavior matrix having the Boolean system and the 4 manual ad hoc systems, and the second behavior matrix having the Boolean system and the 4 automatic ad hoc systems.

Note that a system with a capability of retrieving many unique documents is not necessarily a better system in terms of precision and recall. A system with such a behavior is merely a different system, and it may be helpful to the searchers in their recall-oriented research work.

3.0 Results

3.1 Retrieval Effectiveness

The one-way ANOVA test results from analyzing the precision-recall matrix suggest that there is no statistical difference among the 10 systems in terms of either precision measure or recall measure [Table 2]. The ANOVA test was repeated with the data transformed first using square root function and second using the inverse trigonometric sine function to take care the potential problem of unequal variances usually associated with percentage figures. The results from the repeated test do not suggest anything new. The results from multivariate analysis of variance that took both precision and recall into account also suggest that there is no statistical difference among the 10 systems.

CLASS	**LEVELS**				
system	**10**				
Dependent:	precision				
Source	**DF**	**Anova SS**	**mean sqr**	**F value**	**Pr > F**
system	9	.507059	.056340	.80	.6198
Dependent:	recall				
Source	**DF**	**Anova SS**	**mean sqr**	**F value**	**Pr > F**
system	9	.131306	.014589	.39	.9408
Manova test:	no system effect				
		S = 2	M = 3	N = 238.5	
Statistic	**Value**	**F**	**Num DF**	**Den DF**	**Pr > F**
Wilks'	.982010	.4853	18	958	.9649
Roy's	.016554	.8829	9	480	.5404

TABLE 2. ANOVA analysis on variable Precision and variable Recall

The 2 Tukey Groupings generated by Tukey's Studentized Range Test for the variable precision and the variable recall contain only one group that covers every system, suggesting, again, that there is no significant difference among the 10 systems. The mean precisions and the mean recalls copied from the 2 Tukey Groupings are listed in [Table3] for the 10 systems.

System	N	Precision	Recall	System	N	Precision	Recall
CnQst2	49	.2763	.4672	uwgcl1	49	.2305	.4199
INQ202	49	.2489	.4226	Boolean	49	.2281	.4139
pircs2	49	.2431	.4000	cityal	49	.2223	.3690
CrnlAE	49	.2404	.4026	INQ201	49	.2201	.3481
CrnlAL	49	.2329	.3951	pircs1	49	.2195	.3676

TABLE 3. Mean precisions and Mean recalls from the 2 Tukey's tests

3.2 Retrieval Behavior

The results from analyzing the retrieval behavior of the Boolean and automatic ad hoc systems suggest that the systems are different. As illustrated in [Table 4] some systems in the first behavior matrix retrieved more uniquely relevant documents as well as more unique documents than other systems in the group. The examination to [Table 5] reveals that it is the Boolean system that made the difference. The Boolean system is statistically different from any automatic ad hoc system in the group in retrieving more uniquely relevant documents. Read the mean column in [Table5] for a quick review.

CLASS **system**	**LEVELS** **5**				
Dependent:	UNIQUELY REL				
Source	**DF**	**Anova SS**	**mean sqr**	**F value**	**Pr > F**
system	4	1827.73	456.93	8.91	.0001
Dependent:	UNIQUE				
Source	**DF**	**Anova SS**	**mean sqr**	**F value**	**Pr > F**
system	4	15677.41	3919.35	2.76	.0284
Manova Test:	no system effect				
		S = 2	M = 0.5	N = 118.5	
Statistic	**Value**	**F**	**Num DF**	**Den DF**	**Pr > F**
Wilks'	.854631	4.8821	8	478	.0001
Roy's	.149153	8.9492	4	240	.0001

TABLE 4. ANOVA analysis on variable Uniquely Relevant Document and variable Unique Document in the first behavior matrix

Alpha = 0.5	**DF = 240**	**MSE = 51.27**	
Critical value of studentized range = 3.887	**Minimum significant difference = 3.976**		
Means with the same letter are not significantly different			
Tukey Grouping	**Mean**	**N**	**System**
A	9.612	49	boolean
B	3.837	49	CrnlAE
B	3.224	49	pircs1
B	2.714	49	cityal
B	2.020	49	INQ201

TABLE 5. Tukey's Studentized range test for variable Uniquely Relevant Document

For the variable unique document, a similar examination to [Table 6], however, reveals a slightly different picture. It is that the Boolean system only retrieved more unique documents than "CrnlAE" did, the Boolean system retrieved a similar number of unique documents as the three other systems, namely, "INQ201" "cityal" and "pircs1."

Alpha = 0.5		**DF = 240**	**MSE = 1419.1**	
Critical value of studentized range = 3.887		**Minimum significant difference = 20.919**		
Means with the same letter are not significantly different				
Tukey Grouping		**Mean**	**N**	**System**
	A	39.878	49	boolean
B	A	25.939	49	INQ201
B	A	22.061	49	cityal
B	A	21.776	49	pircs1
B		16.122	49	CrnlAE

TABLE 6. Tukey's Studentized range test for variable Unique Document

Turning to the second behavior matrix consisting of the Boolean system and the manual ad hoc systems, the picture is more complicated than the one described in the preceding paragraph for the automatic ad hoc systems. As before the results from the general one-way ANOVA analysis reported in [Table 7] for the variable uniquely relevant document and the variable unique document suggest that there is at least one system in the current matrix that is significantly different from the rest.

The voting in multiple comparison in [Table 8] is that the Boolean system retrieved more uniquely relevant documents than “INQ202” did, and both “uwgcl1” and “CnQst2” retrieved more uniquely relevant documents than “INQ202” did. “pircs2” is apparently neutral and is indifferent from any other system. In terms of retrieving just unique documents the Tukey grouping in [Table 9] fails in indicating which system(s) contributed to the significant difference detected in [Table 7]. It is only the pairwise T-tests in [Table 10] that suggest that the Boolean system is more productive in finding unique documents than “pircs2” and “INQ202”, and “uwgcl1” is more productive than “INQ202”. The tests in [Table 9-10] still suggest that the Boolean system is different from at least some of the manual ad hoc systems in retrieving unique documents for the TREC4 ad hoc queries.

CLASS	**LEVELS**				
system	**5**				
Dependent:	UNIQUELY REL				
Source	**DF**	**Anova SS**	**mean sqr**	**F value**	**Pr > F**
system	4	1000.68	200.136	4.29	.0009
Dependent:	UNIQUE				
Source	**DF**	**Anova SS**	**mean sqr**	**F value**	**Pr > F**
system	4	10719.82	2143.964	2.43	.0353
Manova Test:	no system effect				
		S = 2	M = 1	N = 142.5	
Statistic	**Value**	**F**	**Num DF**	**Den DF**	**Fr > F**
Wilks’	.918748	2.4844	10	547	.0064
Roy’s	.079987	4.6073	5	288	.0005

TABLE 7. ANOVA analysis on variable Uniquely Relevant Document and variable Unique Document in the second behavior matrix

			Alpha = 0.5	DF = 240	MSE = 46.676
Critical value of studentized range = 4.057			Minimum significant difference = 3.96		
Means with the same letter are not significantly different					
Tukey Grouping			Mean	N	System
	A		6.429	49	boolean
B	A		6.245	49	uwgcl1
B	A		6.082	49	CnQst2
B	A	C	3.735	49	pircs2
		C	2.082	49	INQ202

TABLE 8. Tukey's Studentized range test for variable Uniquely Relevant Document

	Alpha = 0.5	DF = 240	MSE = 882.37
Critical value of Studentized range = 4.057	Minimum significant difference = 17.218		
Means with the same letter are not significantly different			
Tucky Grouping	Mean	N	System
A	32.020	49	boolean
A	30.796	49	uwgcl1
A	24.245	49	CnQst2
A	19.000	49	pircs2
A	16.959	49	INQ202

TABLE 9. Tukey's Studentized range test for variable Unique Document

Alpha = 0.5; Confidence = 0.95; DF = 240 MSE = 883.37

Critical value of T = 1.96824

Least significant difference = 11.812

Comparisons significant indicated by ***	**Difference between Means**	
boolean - uwgcl1	1.224	
boolean - CnQst2	7.776	
boolean - pircs2	13.020	***
boolean - INQ202	15.061	***
uwgcl1 - CnQst2	6.551	
uwgcl1 - pircs2	11.796	
uwgcl1 - INQ202	13.837	***

.... no other comparisons are significant

TABLE 10. T test (LSD) on variable Unique Document

4.0 Conclusion

A study of comparing the Boolean retrieval system to the general ranking systems was carried out during and after TREC4. The conference provided a rare opportunity for constructing a sound experiment design to compare these two types of retrieval systems. The design benefited from 1) the large and diverse test databases, 2) the short query topics, 3) the interactive search instructions for Boolean search, and 4) the list of the best performed ranking systems for comparison. The design used the existing cutoff method, based on the size of a Boolean document set to turn the ranking document list to a document set for comparing it to the Boolean set. This method separates retrieval and presenting since the former can be studied statically while the latter needs a situation of information need to justify the effectiveness of a particular presentation scheme such as subject relevance or subject currency. The discussion of the design suggests that for the purpose of presenting retrieved documents subject relevance is merely one of many different relevance dimensions, and that with the possibility of presenting the retrieved along different relevance dimensions, the traditional relevance ranking principle appears rigid and inflexible.

The test results from this static design suggest that the Boolean system could have a similar performance as the best relevance ranking systems in the environment of TREC4. A closer examination on mean precision and mean recall, however, suggest that the selected ranking systems, especially the manual ad hoc systems did slightly better, but not statistically significant. The results from the one-way ANOVA tests and the associated MANOVA tests are very consistent throughout.

Unlike the other reported research work, this study took a step further to examine the retrieval behavior of the selected systems. The behavior was defined as the capability of a system in retrieving uniquely relevant documents and unique documents for a query. The Boolean searcher may hesitate to only rely on the ranking system if the Boolean system maintains such a capability. The results from the behavior tests do suggest that the Boolean system tends to retrieve a set of relevant documents that is significantly different from the relevant set retrieved by the selected ranking systems. On the other hand, the unique document set that the Boolean system tends to create is not that distinct from the unique set that the ranking systems tend to build, especially when the ranking systems are those manual ad hoc systems in TREC4.

5.0 References

[1] Aitchison, T. M. Et al. (1970) "Comparative evaluation of indexing languages, Part II: Results," Report R70/2, INSPEC, Institution of Electrical Engineers, London.

[2] Cleverdon, C. W. (1970) An investigation into a suitable mechanised information retrieval system at the defense operational analysis establishment, Cranfield Institute of Technology.

[3] Miller, W. L. (1971) "The efficiency of MEDLARS titles for retrieval," Journal of the American Society for Information Science, Vol.22, 318-321.

[4] Miller, W. L. (1971) "A probabilistic search strategy for MEDLARS," Journal of Documentation, Vol.27, 254-266.

[5] Salton, G. (1972) “A new comparison between conventional indexing (MEDLARS) and automatic text processing (SMART),” Journal of the American Society for Information Science, Vol.23, 75-84.

[6] Evans, L. (1975) “Search strategy variations in SDI profiles,” Report R75/21, INSPEC, Institution of Electrical Engineers, London.

[7] Evans, L. (1975) “Methods of ranking SDI and IR outputs,” Report R75/23, INSPEC, Institution of Electrical Engineers, London.

[8] Turtle, H. (1994) “Natural language vs. Boolean query evaluation: A comparison of retrieval performance,” Proceedings of the 17th Annual International ACM-SIGIR conference, 212-221, Dublin, Ireland.

[9] Harman, D. K., editor. (1996) Overview of the Fourth Text REtrieval Conference (TREC-4), to be published by NIST.

[10] Lu, X. A. and Keefer, R. B. (1995) “Query expansion/reduction and its impact on retrieval effectiveness,” Overview of the Third Text REtrieval Conference (TREC-3), NIST Special Publication 500-225, edited by D. K. Harman, 231-240.

[11] Tague, J. M. (1981) “The pragmatics of information retrieval experimentation,” Information Retrieval Experiment, Chapter 5, 59-102, edited by K. S. Jones, Butterworth.

Improvements on Query Term Expansion and Ranking Formula

Kenji Satoh, Susumu Akamine and Akitoshi Okumura
Human Language Research Lab. Information Technology Research Labs.
NEC Corp.

Abstract

In TREC-3, we developed an information retrieval system which supported automatic query term generation by syntax parser and document matching by vector space model.

In TREC-4, we improved the following two points according to the evaluation result of TREC-3.

Query Term Expansion: A query is enhanced by a thesaurus and a word co-occurrence data base. Useless general words are eliminated by the standard deviation of the word frequencies. Routing queries were expanded by the word co-occurrence data base, and ad-hoc queries are expanded by a WordNet thesaurus, when we submitted the results to NIST on August.

Ranking Formula: The ranking formula is improved by using the summation of term weights in each document instead of the internal product of term weights and the word frequencies. Values for internal product factors are determined on the basis of several tests.

1. Introduction

As we reported at the TREC-3 conference last year, we developed an information retrieval system within one month only by three people. This system used query term generation with syntax parser and document matching with vector space model. When we evaluated the system, we found the following two problems:

a) Noun phrases extracted from TOPICs are insufficient to retrieve all relevant documents.

b) Document ranking calculation uses internal product of word frequency and term weight, and the frequency is often very large especially in long documents.

We have improved these two points.

Query Term Expansion: A query is enhanced by a thesaurus and a word co-occurrence data base. Useless general words are eliminated by the standard deviation of the word frequencies. Routing queries were expanded by the word co-occurrence data base, and ad-hoc queries are expanded by a WordNet thesaurus, when we submitted the results to NIST on August.

Ranking Formula: The ranking formula is improved by using the summation of term weights in

each document instead of the internal product of term weights and the word frequencies. Values for internal product factors are determined on the basis of several tests.

Figure 1 shows the system modules. The word co-occurrence extracted from all the documents is used for query term expansion in the routing task phase, and WordNet thesaurus is used in the ad-hoc task phase.

This paper describes the system in the order of Query Term Expansion, Ranking Formula and whole system processes.

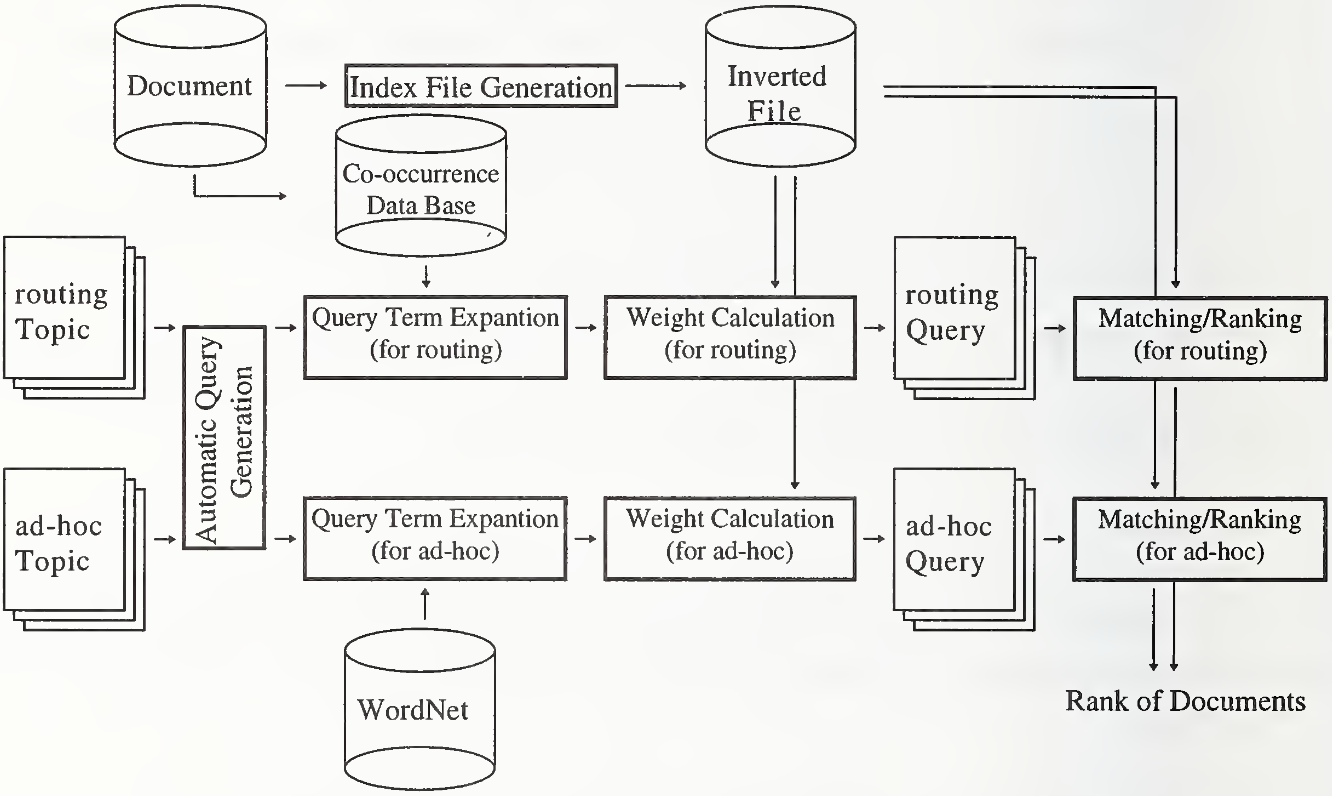

Document
Index File Generation
Inverted File
Co-occurrence Data Base
routing Topic
Automatic Query Generation
Query Term Expantion (for routing)
Weight Calculation (for routing)
routing Query
Matching/Ranking (for routing)
ad-hoc Topic
Query Term Expantion (for ad-hoc)
Weight Calculation (for ad-hoc)
ad-hoc Query
Matching/Ranking (for ad-hoc)
WordNet
Rank of Documents

Figure 1: System Modules

2. Query Term Expansion

Queries are automatically created by two processes, initial query generation and query term expansion. Initial query terms are a set of noun phrases extracted from TOPICs by the syntactic parser. The query terms are expanded by the co-occurrence data base and thesaurus.

2.1 Expansion by co-occurrence data base

Word co-occurrence is used for query term expansion in the routing task phase. Co-occurrence data base created based on the word sets co-occurring in the same paragraph. All TREC documents are used to create the data base.

Co-occurrence data base includes both closely-related words and loosely-related words. For example, ‘computer' appears in the co-occurrence data base as the most frequent word because ZIFF data includes only computer topic. Very general words such as `computer' are rather harmful than useful for retrieval, because they resulted in the irrelevant document retrieval when the number of query terms is small. It is therefore necessary to select the distinctive words for the term expansion.

Because relevant weight models based on relevance judgment can be used in the routing task phase, query terms are expanded by co-occurrence information. We also select only proper nouns from expanded words, because proper nouns are important for retrieving documents. Proper nouns are selected on the basis of their beginning with an upper-case letters.

2.2 Expansion by thesaurus

We used WordNet as thesaurus for the ad-hoc task phase. WordNet provides a word with synonyms, hyponyms, and other related information according to the semantic entries. Query terms with one semantic entry are expanded by the synonyms. Query terms with several semantic entries are not expanded because the expansion might add irrelevant words to the query. We also determine a threshold for the standard deviation of word frequencies to avoid including general words.

3. Ranking Formula

In TREC-3 system, rankings were calculated on the basis of the internal product of word frequency and term weights. The value of the internal product was often very large especially in long documents, because word frequency was very large in these documents even if a generalized factor of the document length was applied. We thus revised the generalized factor of the document this year.

A relevant weight model was introduced into the routing task phase. The model uses a summation of the term weights in each query which appears in the document is used as the means of calculation.

The standard deviation of word frequencies is used for ad-hoc query term weights. Through applying several parameters to the word frequencies, we found that `logarithm' and `square root' functions lead to the opposite results of the TREC-3 system, i.e. the internal product of short documents was often very large. Because constant parameters produced better results than these functions, we determined a value for these parameters on the basis of several tests.

4. System processing

4.1 Routing Task

Figure 2 shows a system processing of the routing task phase.

A. Query term generation

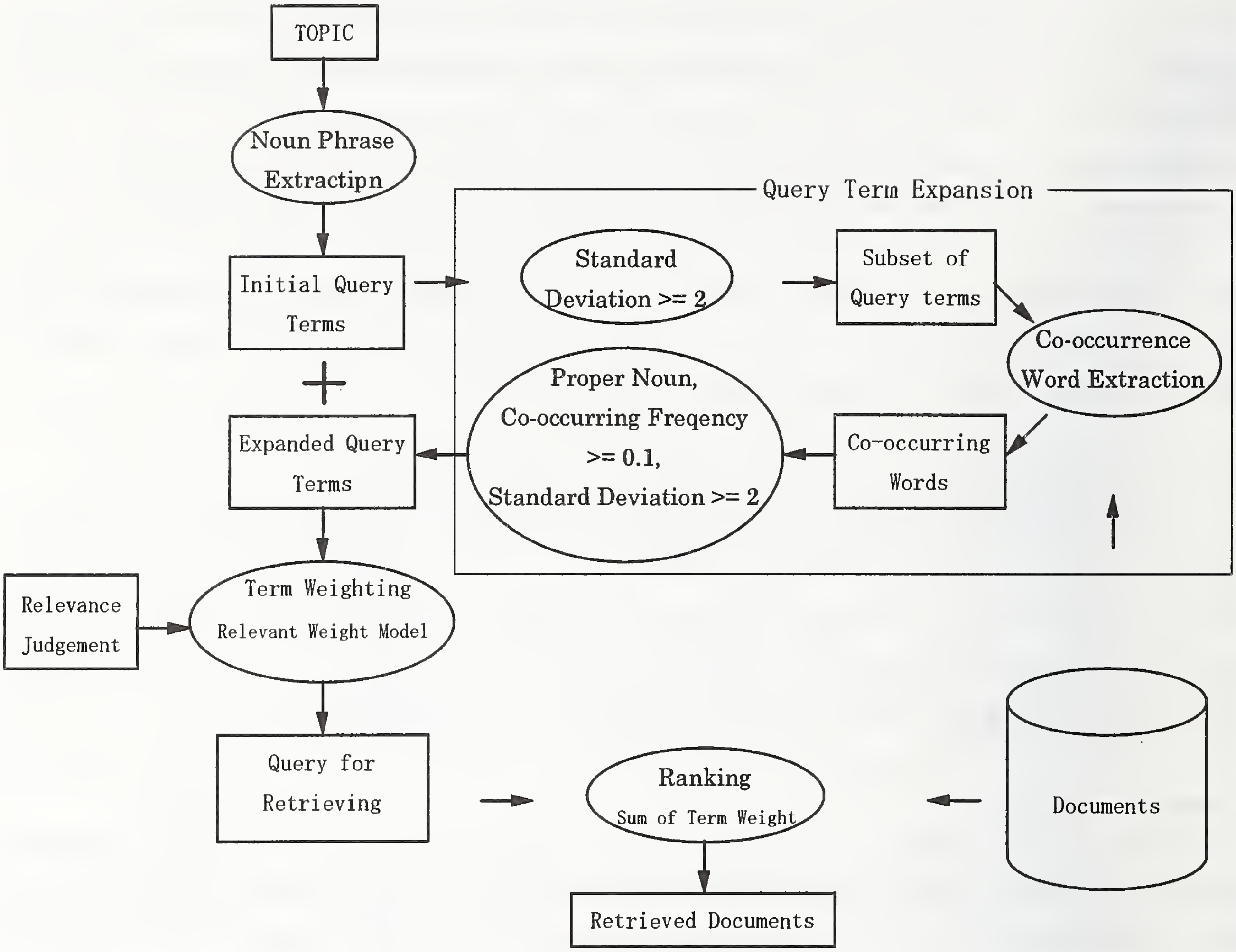


Figure 2: Routing Process

1) Initial query terms are noun phrases which were extracted from TOPICs.

2) Co-occurrence information is calculated in the TREC documents with initial query terms which were selected on the basis of having a standard deviation value larger than 2.0.

3) Or-set of co-occurrence information of each query is calculated. (Co-occurring frequency is averaged.)

4) Proper nouns (beginning with an upper-case letter) are selected from 3).

5) Expansion words are selected on the basis of having a co-occurring frequency larger than 0.1.

6) Expansion words selected because the standard deviation of their own frequency is larger than 2.0 are added to the initial query terms.

B. Query term weighting

Relevant weight model using relevance judgment

$$w_i = \frac{(r_i + 0.5)/(R - r_i + 0.5)}{(n_i - r_i + 0.5)/(N - n_i - R + r_i + 0.5)}$$

N : the number of documents

n_i : the number of documents containing the term i

R : the number of relevant documents

r_i : the number of relevant documents containing the term i

C. Ranking formula

Summation of the weights of terms appearing in each document

$$S_i = \sum_{j=1}^{N} w_{ij}$$

S_i : the score of document i

w_{ij} : the weight of term j appearing in document i

4.2 Ad-hoc Task

Figure 3 shows a system processing of the ad-hoc task phase.

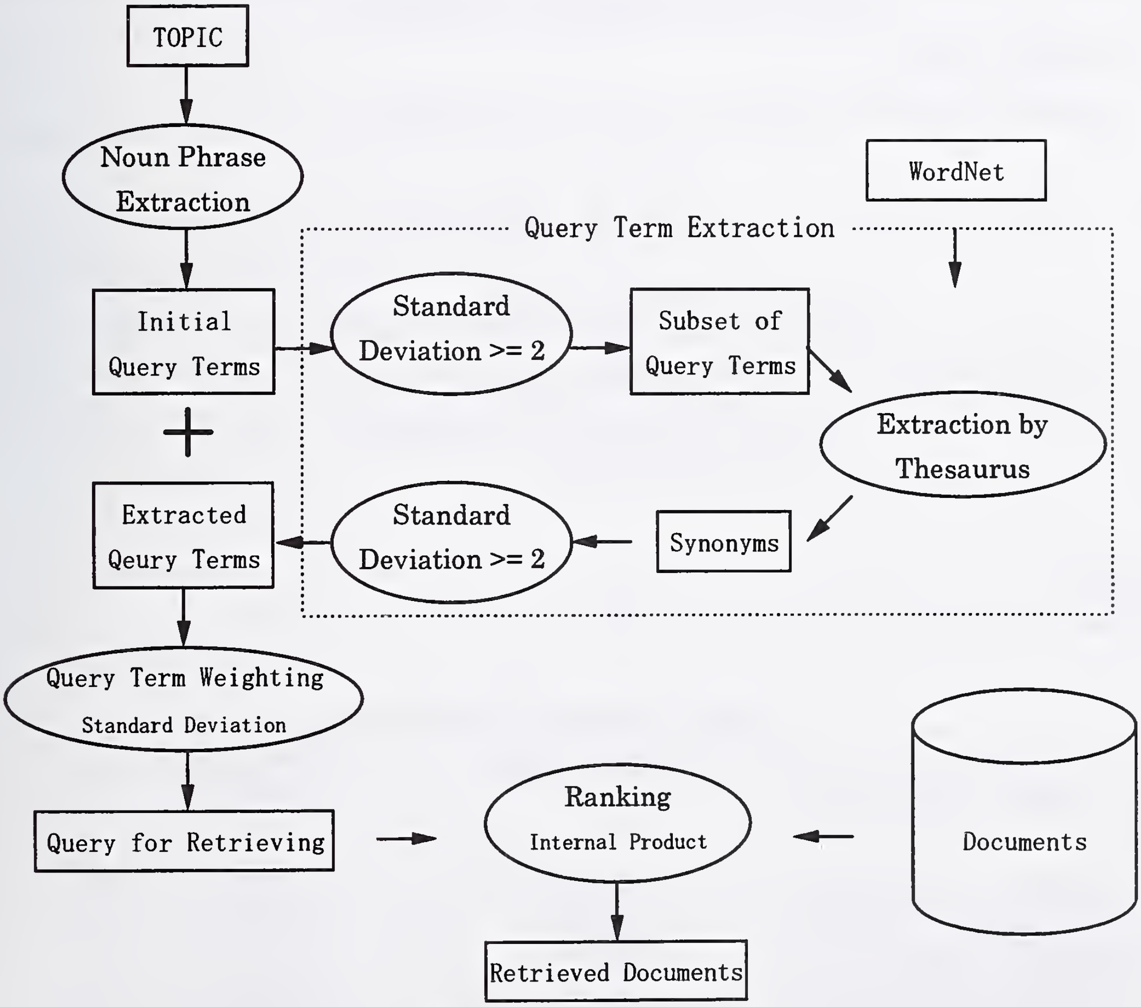


Figure 3: Ad-hoc Process

A. Query term generation

1) Initial query terms are noun phrases extracted from TOPICs.

2) Expansion words are extracted by WordNet from initial query terms which have a standard deviation larger than 2.0. Thus query terms with one semantic entry are selected.

3) Expansion words selected because the standard deviation of their own frequency is larger than 2.0 are added to the initial query terms.

B. Query Term weighting

Standard deviation of word frequency in TREC documents

$$w_i = \frac{N}{n_i}\sqrt{\frac{1}{L}\sum_{j=1}^{L}\left(\frac{m_{ij}}{M_j}-\frac{n_i}{N}\right)^2}$$

L : the number of documents

N : the total number of terms in all documents

n_i : the number of term i in all documents

M_j : the total number of terms in document j

m_{ij} : the number of term i in document j

C. Ranking formula

Internal product of word frequency and query term weights

$$S_i = \sum_{j=1}^{N}\left(\frac{d_{ij}}{D_i+100} * w_i\right)$$

S_i : the score of document i

D_i : the total number of terms in document i

N : the total number of terms

d_{ij} : the number of term j in document i

w_j : the weight of term j

5. Discussion

The results of our system are shown in Table 1.

Table 1: Results of our systems

		Average Precision	R-Precision
routing	TREC-3(manual)	0.2717	0.3204
	TREC-4(automatic)	0.2270	0.2637
ad-hoc	TREC-3(automatic)	0.0624	0.1170
	TREC-4(automatic)	0.1082	0.1539

The difference between results of TREC-3 and TREC-4 in routing task phase comes from the methods of query term generation. Though we could not keep quality of query terms with automatic query term generation, query term expansion with co-occurrence information is not so bad, it is necessary to select query terms with other information.

The difference between results of TREC-3 and TREC-4 in ad-hoc task phase seems to come from query term expansion by WordNet. Though the result of TREC-4 is better, it is necessary to consider query term expansion by meronymy words in WordNet (as well as synonyms), and to compare other thesauruses with the WordNet.

We also need to examine other methods of calculating co-occurrence information, such as the word frequency in the same sentence or within adjacent words, rather than words in the same paragraph.

6. Conclusion

We have improved our retrieval system from two aspects, query term expansion and ranking formula.

Unfortunately, we had to submit data to NIST without trying lots of promising methods because the programs took two or three days for the calculation of co-occurrence information from TREC documents.

We are improving the following methods:

1) The use of words other than proper noun after the extraction of co-occurrence words
2) The use of other thesaurus than WordNet
3) The combined use of a thesaurus and word co-occurrence information
4) The creation of a query term expansion knowledge base by using relevance judgment to evaluate expanded terms, and then repeating the expansion several times on the basis of those results

A TREC EVALUATION OF QUERY TRANSLATION METHODS FOR MULTI-LINGUAL TEXT RETRIEVAL

Mark Davis and Ted Dunning
{madavis, ted}@crl.nmsu.edu
Computing Research Lab
New Mexico State University
Box 30001/3CRL
Las Cruces, NM 88003

ABSTRACT

In a Multi-lingual Text Retrieval (MLTR) system, queries in one language are used to retrieve documents in several languages. Although all of the collection documents could be translated to a single language, a more efficient approach is to simply translate the queries into each of the document languages. We have investigated five methods for query translation that rely on lexical-transfer and corpus-based methods for creating multi-lingual queries. The resulting queries produced by these systems were then used in a competitive information-retrieval environment and the results evaluated by the TREC evaluation group.

INTRODUCTION

The goal of text retrieval is to retrieve documents that are closely related to a user's needs. In general, the problem of translating the user's needs into queries for text retrieval systems is central to the field of text retrieval (Salton, 1970). The query may contains many more words than appear in the original user-input. In addition, users are generally rather poor at assigning weights to the terms in the query. Choosing good terms and weighting them is hard enough in one language, but when the problem is extended to involve documents in multiple languages it becomes considerably more difficult. In one possible scenario, the user generates input for the system in only a single language, but expects to retrieve documents in multiple languages (multi-lingual text retrieval or MLTR). In the context of conventional vector-based retrieval systems, this could be accomplished by automatically translating all of the documents into a single language when indices of the documents are created, or by translating the user-input into a multi-lingual query.

This paper describes the results of work on several approaches to MLTR that build multi-lingual queries using a lexicon and a corpus of previously translated documents. The comparative performance of the different methods is evaluated using original Spanish queries run with a competitive TREC text-retrieval engine as a baseline. The query translation process then operates on hand-translated versions of the original queries and the resulting translations are run with the same system.

A BRIEF HISTORY OF MULTI-LINGUAL TEXT RETRIEVAL

Salton (1970) first demonstrated that text retrieval could be used in a multi-lingual setting. His system used a thesaurus for generating query translations by taking the terms in the thesaurus for each query term and forming a new translated query. The thesaurus was created by hand for the retrieval corpus and the entries were therefore inherently disambiguated with respect to the corpus domain prior to query generation. Nevertheless, Salton's results demonstrate that IR systems can perform well in a multi-lingual setting using simple translation resources. Unfortunately, domain-specific, up-to-date glossaries are generally difficult to obtain. Those that are produced are typically constructed by and for translators, who write them in the process of translation, suggesting that an approach which makes use of the translations directly in combination with other resources is needed.

Experiments with latent semantic indexing (LSI) (Landauer and Littman, 1990) showed that paragraphs which were translations of each other could be retrieved but no actual retrieval system was constructed, nor was it clear how the system would perform in practice. This use of parallel corpora eliminates many of the problems of using bilingual dictionaries, but introduces new problems. In particular, in the context of a traditional vector based retrieval system, it has not been clear how to perform multi-lingual retrieval based on the information contained in parallel translated corpora. The success of experiments with LSI does not directly provide a method to make a more traditional vector based system work. Furthermore, LSI makes the use of inverted indexes problematic, which may hinder the practicality of this system.

Dunning and Davis (1993a,b) developed a system for multi-lingual text retrieval based on a novel method for solving very large systems of linear equations. In this system, query translation was viewed as a linear transformation of a query feature vector. For long strings, the translation of the concatenation of the strings is approximately the translation of the strings independently. This is true because the translation of two strings is nearly the concatenation of their translations. While this linearity breaks down dramatically at the word level, at the sentence level and above, it works fairly well. Despite the simplification afforded by linearity in the transformation, the actual translation matrix was derived through a computationally-taxing error minimization strategy that used a parallel aligned corpus of 50,000 words as exemplars to iteratively update the transformation matrix. At that time, machine resources were very limited and the algorithm had poor convergence properties.

Davis and Dunning (1995) applied an evolutionary approach to parallel collections by optimizing for translated query performance over a collection of parallel texts. The initial queries were created from term-by-term lookup in a bilingual machine readable dictionary. Although promising, the work left many unanswered questions about the usefulness of general lexicons for highly specific text domains and the value of corpus-based term disambiguation in an IR framework. Moreover, the results appeared to be affected by the lack of high-quality parallel text for training and evaluation.

NEW APPROACHES TO MLTR

Starting with TREC- 3, a Spanish corpus and query sets have been available for evaluating text retrieval engines. The queries and corpus are monolingual, however, so testing a multi-lingual system is only possible if the query set or the corpus is translated into a different language. We chose to translate the queries since they were very short. With translated queries, a query translation system that produces Spanish queries from hand-translated English versions of original Spanish queries can then be compared against the original queries. The differences between the two results are then a reasonable measure of the effectiveness of the translation process in preserving the characteristics of the original query that contribute to retrieval. This assumes, of course, that the English hand translation preserves the retrieval characteristics and remains an unknown in our evaluation efforts. The Spanish TREC queries and their hand-translated versions are shown in Table 1, below.

Table 1: Spanish TREC queries and English Translations

Q#	Original Spanish Queries	Hand-translated English
26	Indicaciones de las relaciones económicas y comerciales de México con los paises europeos.	Indicators of economic and business relations between Mexico and European contries.
27	Indicaciones de las relaciones económicas y comerciales de México con los paises africanos.	Indicators of economic and business relations between Mexico and African contries.
28	Indicaciones de las relaciones económicas y comerciales de México con los paises asiáticos, por ejemplo Japon, China y Corea.	Indicators of economic and business relations between Mexico and Asian countries, such as Japan, China and Korea.
29	Indicaciones de las relaciones económicas y comerciales entre México y Canadá.	Indicators of economic and business relations between Mexico and Canada.
30	¿Hay programas y intercambios deportivos entre México y los Estados Unidos?	Are there sports programs and exchanges between Mexico and the United States?
31	¿Cuáles son Ias medidas tomadas por el gobierno mexicano para resolver la disputa con los rebeldes zapatistas en el estado de Chiapas?;	What measures has the Mexican government taken to resolve the quarrel with the rebel Zapatitas in the state of Chiapas?
32	¿Cuál es la importancia de las Naciones Unidas (NU) para México?	What importance does the United Nations (UN) have for Mexico?
33	¿Cómo van las relaciónes entre México y la Organización de los Estados Americanos (OEA)?	How are relations between Mexico and the Organization of American States (OAS)?
34	Indicaciones de los potenciales y debilidades del ejército mexicano.	Indicators of the Mexican Army's strengths and weaknesses.
35	Indicaciones de los potenciales y debilidades de las fuerzas aéreas militares de México.	Indicators of the Mexican Air Force's strengths and weaknesses.
36	Indicaciones de los potenciales y debilidades de la marina de guerra (fuerzas navales, armada) de México.	Indicators of the Mexican Navy's (naval and marine forces) strengths and weaknesses.
37	Evidencia de la herencia y cultura azteca en México.	Evidence of Aztec heritage and culture in Mexico.
38	¿Hay programas para la renovación urbana en México?	Are there urban renewal programs in Mexico?
39	¿Cuáles son Ias medidas modernas para mejorar la agricultura en México?	What modern measures for agricultural impovement are there in Mexico?
40	Información sobre el ballet folklórico en México.	Information about Mexico's traditional dance (ballet folklorico).
41	Medidas para controlar o evitar inundaciones en México.	Flood prevention and control measures in Mexico.
42	¿Tendrá éxito el NAFTA (TLC) en México?	Will NAFTA (TLC) be successful in Mexico?
43	Hay programes para reprimir o limitar epidemias en México?	Are there epidemic control programs in Mexico?
44	Información sobre la industria de computadoras mexicana.	Information about Mexico's computer industry.
45	Actitudes en México sobre la censura de la prensa.	Attitudes in Mexico regarding censorship of the press?
46	Informes sobre visitas oficiales y privadas a México por jefes de estado y de gobierno.	Reports regarding official and private visits to Mexico by chiefs of state and heads of government?
47	Hay programs en México para investigar la causa de cáncer?	Does Mexico have research programs for the cause of cancer?
48	Hay programas internacionales para el intercambio de estudiantes en México?	Are there international student exchange programs in Mexico?
49	El turismo como fuente de divisas para México.	Tourism as a source of Mexico's income.
50	La fabricación en México de joyas de plata y oro.	Silver and gold jewelery manufacturing in Mexico.

The query translation methods that we applied to produce new Spanish queries were of two major types: methods that used a prepared lexicon and methods that used a parallel training corpus. While a lexicon tends to produce translations that are shallow but comprehensive, covering all possible senses of a term but limited in the range of synonyms that are produced for each term, corpus methods tend to produce translations that are deep but narrow, with enormous repetition of domain-related senses of terminology. This justified an examination of the comparative merits of both approaches.

As is often the case, our parallel corpus was not precisely of the same domain as the TREC document collection for the ultimate evaluation. The corpus itself was extremely large, however, which we hoped would offset the difficulties of using a distinctly different type of text. The corpus was 1.6 Gb of Spanish and English translations from the United Nations, containing proceedings of meetings, policy documents and notes on UN activities in member countries. The documents were automatically aligned (Davis, Dunning and Ogden, 1995) at the sentence level using a procedure that is conservatively estimated to have an 83% accuracy over grossly noisy document pairs (which the UN documents were not). This produced a parallel corpus of around 680,000 aligned sentence pairs.

Lexical Transfer

The first method was to perform term-by-term translation with the Collins English-Spanish bilingual dictionary. Individual terms in the English query were reduced to their morphological roots and lookup was performed. The resulting set of Spanish terms became the Spanish query. Some repetition of terms is apparent in the resulting queries because all senses of each term were used with no attempt to disambiguate the contextual usage of the English terms. For example, Query 28 is transformed from

```
Indicators of economic and business relations between Mexico and Asian coun-
tries, such as Japan, China and Korea.
```

to

```
indicador indicador ayuda expansión previsiones crecimiento comercio comer-
cio narración relación parentesco México Ciudad gripe patria campo región
amor semejante parecido tanto el laca China Mar té porcelana vitrina coalín
Corea Corea Corea mexicana mexicano México
```

Note that “China” has been replaced with both “China” and “porcelana” as a result of this simple lexical substitution scheme, and that “relations” has included the familial sense “parentesco”. The complete set of queries is listed in Table 1.

Table 2: Lexicon-generated Spanish Queries

Q#	Hand-translated English	Lexicon-Generated Spanish
26	Indicators of economic and business relations between Mexico and European contries.	indicador indicador ayuda expansión previsiones crecimiento comercio comercio narración relación parentesco Méjico Ciudad Comunidad Comisión ribunal
27	Indicators of economic and business relations between Mexico and African contries.	indicador indicador ayuda expansión previsiones crecimiento comercio comercio narración relación parentesco Méjico Ciudad
28	Indicators of economic and business relations between Mexico and Asian countries, such as Japan, China and Korea.	indicador indicador ayuda expansión previsiones crecimiento comercio comercio narración relación parentesco Méjico Ciudad gripe patria campo región amor semejante parecido tanto el laca China Mar té porcelana vitrina coalín Corea Corea Corea

Q#	Hand-translated English	Lexicon-Generated Spanish
29	Indicators of economic and business relations between Mexico and Canada.	indicador indicador ayuda expansión previsiones crecimiento comercio comercio narración relación parentesco Méjico Ciudad
30	Are there sports programs and exchanges between Mexico and the United States?	allá deporte caza deporte juego cambio canje intercambio intercambio Méjico Ciudad estado estado
31	What measures has the Mexican government taken to resolve the quarrel with the rebel Zapatitas in the state of Chiapas?	medida medida excesivamente mejicano gobierno administración Estado régimen resolución resolución reyerta cabecilla estado estado
32	What importance does the United Nations (UN) have for Mexico?	importancia nación des no nada poco aciones poseer Méjico Ciudad
33	How are relations between Mexico and the Organization of American States (OAS)?	narración relación parentesco Méjico Ciudad especialista hule inglés fútbol estado estado
34	Indicators of the Mexican Army's strengths and weaknesses.	indicador indicador mejicano resistencia fuerzas intensidad fuerza flojedad tenuidad flaco desventaja
35	Indicators of the Mexican Air Force's strengths and weaknesses.	indicador indicador mejicano resistencia fuerzas intensidad fuerza flojedad tenuidad flaco desventaja
36	Indicators of the Mexican Navy's (naval and marine forces) strengths and weaknesses.	indicador indicador mejicano escuela arquitectura marítimo ingeniero ingeniería seguro fuerza fuerza resistencia fuerzas intensidad fuerza flojedad tenuidad flaco desventaja
37	Evidence of Aztec heritage and culture in Mexico.	evidencia testimonio hechos herencia patrimonio cultura choque choque Méjico Ciudad
38	Are there urban renewal programs in Mexico?	allá zona éxodo guerrillero renovación renovación reanudación extensión prorrogación Méjico Ciudad
39	What modern measures for agricultural impovement are there in Mexico?	medida medida excesivamente agropecuario escuela-granja perito feria allá Méjico Ciudad
40	Information about Mexico's traditional dance (ballet folklorico).	información noticias aproximadamente baile danza ballet
41	Flood prevention and control measures in Mexico.	inundación avenida pleamar torrente prevención medida medida excesivamente Méjico Ciudad
42	Will NAFTA (TLC) be successful in Mexico?	Méjico Ciudad
43	Are there epidemic control programs in Mexico?	allá control control dominio Méjico Ciudad
44	Information about Mexico's computer industry.	información noticias aproximadamente ordenador informático industria
45	Attitudes in Mexico regarding censorship of the press?	actitud ademán postura disposición Méjico Ciudad censura presión apretón presa
46	Reports regarding official and private visits to Mexico by chiefs of state and heads of government?	relato parte informe noticia autorizado huelga particular detective visita Méjico Ciudad jefe jerarca jefe estado estado cabeza cabellera gobierno administración Estado régimen
47	Does Mexico have research programs for the cause of cancer?	Méjico Ciudad poseer investigación investigación investigaciones causa causa cáncer canceroso investigación
48	Are there international student exchange programs in Mexico?	allá Cámara Corte línea derecho estudiante investigador asociación alumnado cambio canje intercambio intercambio Méjico Ciudad
49	Tourism as a source of Mexico's income?	turismo el fuente procedencia fuente foco ingresos rédito
50	Silver and gold jewelery manufacturing in Mexico?	abedul abeto hoja oro barra galón oro capacidad costos industrias Méjico Ciudad

The lexical-transfer approach produced Spanish queries rapidly, requiring only a simple database lookup procedure.

High-frequency Terms from a Parallel Corpus

In text, the terms that occur with the highest frequency are rarely of statistical significance, and are more often than not merely redundant. Yet the terms that occur with moderate frequency are sometimes significant. In order to evaluate other corpus-based methods, we wanted to establish a baseline for queries formed from these moderate frequency term sets. Using a vector-based text retrieval system with no term spreading or other modifications, the English queries were translated by performing a lookup on the English side of the parallel corpus, collecting the Spanish sentences that were parallels to the top 100 retrieved documents, filtering the remaining terms to eliminate the top 500 most frequent Spanish terms, and collecting the next 100 most frequent Spanish terms to create the new query. This process is shown in Figure 1, below:

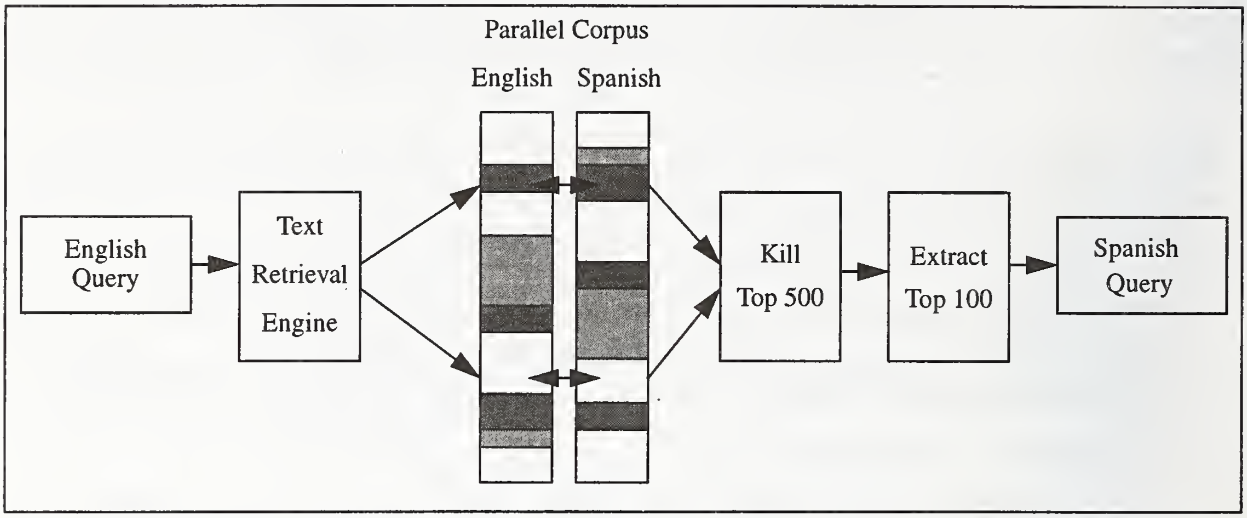


Figure 1. Building Spanish queries by extracting high-frequency terms from a parallel

Several of the resulting queries are given in Table 3. Some formatting codes from the UN documents have been eliminated in some of the queries, reducing the count to below 100 terms in those queries. For brevity, only the first four queries are shown in Table 3.

Table 3: High-frequency Queries from a Parallel Corpus

Q#	Hand-translated English	Corpus High-Frequency Spanish
26	Indicators of economic and business relations between Mexico and European contries.	Checoslovaquia En Ghana Polonia nacional programa Australia Bajos Egipto España Filipinas La Países Portugal Igualdad Italia Paz recursos Austria Finlandia Acción Pide Venezuela Naciones gubernamentales Unidas como período una Comisión Desarrollo regionales sesiones Mujer Mundial información nacionales informe México resolución no proyecto un actividades países Estados organizaciones desarrollo sus su E/CN mujer Secretario General por República al con se Conferencia sobre para del las que los el en la de
27	Indicators of economic and business relations between Mexico and African contries.	Checoslovaquia Democrática Egipto Filipinas Francia Indonesia Irlanda Los Países Secretario Uruguay aplicación más proyectos servicios Alemania Colombia La fuentes trabajo Asamblea Iraq Naciones Nigeria Pakistán Unidos documento han DE Unidas energía nuclear sus Brasil principios siguientes utilización Argentina Chile En Venezuela como desarrollo espacio ultraterrestre El General una período sesiones al países su Estados sobre un para República por con se México que las del los en el la de
28	Indicators of economic and business relations between Mexico and Asian countries, such as Japan, China and Korea.	Italia Repúblicas Chile Ecuador General Gran India Pakistán Suecia Unión Venezuela Votos sus Bretaña Filipinas Norte Nueva Socialistas Soviéticas siguiente Alemania Bajos Colombia España Grecia Indonesia Reino su Argentina Francia Japón Unido especializados Brasil Países sistema Arabe organismos países Nicaragua Sudáfrica China organizaciones contra El México América sobre se Representante ante Irlanda con Permanente Unidos Seguridad para Estados Carta fecha dirigida Presidente Consejo Naciones Unidas al que República por en las los del el la de

Q#	Hand-translated English	Corpus High-Frequency Spanish
29	Indicators of economic and business relations between Mexico and Canada.	Nicaragua Nigeria Uganda Yugoslavia Zelandia con favor Dinamarca Indonesia Norte Ucrania Venezuela Yemen para Australia Botswana Finlandia Italia Soviéticas Suecia Austria Bangladesh Bulgaria Chile Hungría Irán Islámica Noruega Pakistán Repúblicas Socialistas Bajos Brasil Bretaña España Gran Países Reino Unión Unido Egipto Francia Perú Polonia América Argentina India Japón Kenya Canadá Checoslovaquia China Alemania Guinea Nueva Colombia Democrática Votos El Irlanda Arabe Estados México Unidos que las los el del en la República de

Statistically Significant Terms

Whereas the high-frequency terms extracted in the previous method provide a baseline for examining improved methods, high-frequency terms are themselves not necessarily the best terms for discriminating the significant features involved in text retrieval. A better approach is to extract the terms which are statistically significant in the retrieved segments of parallel text in comparison to the corpus as a whole. Various methods are possible for testing statistical significance, but the method we applied is based on a log-likelihood ratio test that assumes a χ^2 distribution is an accurate model of the term distributions in text (Dunning, 1993).

The method begins by extracting all of the terms from the sentences that are parallels to the top 100 retrieved English sentences. The counts of the pooled terms are then compared with the counts for the entire UN training corpus to evaluate their statistical significance. The top 100 most-significant terms are then extracted and become the new Spanish query. Figure 2 diagrams the process. The resulting queries are in Table 4, below.

Table 4: Statistically-significant Spanish Queries

Q#	Hand-translated English	Statistically-significant Spanish Queries
26	Indicators of economic and business relations between Mexico and European contries.	período un una Anguila CARICOM Dos ECCB En Este Oeste Europeo Guyana Jefes Magreb Occidente Parlamento Principal T al ciencias con consentimiento consulares convenciones correo cuantitativos de del diplomáticos el empresarial en experiencias externas guías la Ias los para por que residente se sobre su sustituir tecnológica temporal tienden tomaron tono totalidad trabajan tradicionales transacci transacción transacciones transición transparencia tratará tratase trigésimo trimestre tropiezan trueque ultimado un un Seminario una unificado university urbanas utilizarse véanse vacantes validez vecindad vecinos venían vencimientos vende versión vigentes vinculadas vinculado vinculados voluntarios y Sudáfrica y financiación y rechazó
27	Indicators of economic and business relations between Mexico and African contries.	árboles Anguila CARICOM ECCB En Este Oeste Guyana Jefes Principal al ascenso autóctonos ciencias con consentimiento consulares convenciones correo cuantitativos de del diplomáticos el empresarial en experiencias externas guías la las litorales los mar nato occidental para por que se semillas sobre su títulos tecnológica temporal terremoto tienden tierras titular tomaron tono totalidad trabajan tradicional tradicionales transacción transacciones transición transparencia tratará tratase trimestre tropicales tropiezan trueque un un Seminario una unas unificado urbanas utilizan véanse víctima vecindad vecinos venían vencimientos vende verán versión vigentes vinculadas vinculado vinculados voluntarios vulnerables y Sudáfrica y financiación y rechazó

Q#	Hand-translated English	Statistically-significant Spanish Queries
28	Indicators of economic and business relations between Mexico and Asian countries, such as Japan, China and Korea.	Aprobación Bin Development En Estudio Kumar Nehru Omán Panjaponés Phase Productos Raczkowski Reimnitz Resources Sindicato Trabajadores Water al austríacos centavos chelines computadoras con de del diplomáticas el el en francos la las los mantienen mm navales nombró págs para poblaciones por que revestía sazón se secundario semana semanas siglos sindicatos sobre solo su sucesión sudoriental suizos suizos sujeción sumamente suspendió sustantivas sustituya títulos termina tradicionales transacciones tribales trigésimo un una unificación vacante vecindad venía vicepresidentes vinculadas vio visitaba visitas voz vuelos y Venezuela y el Reino y estudios y financiación

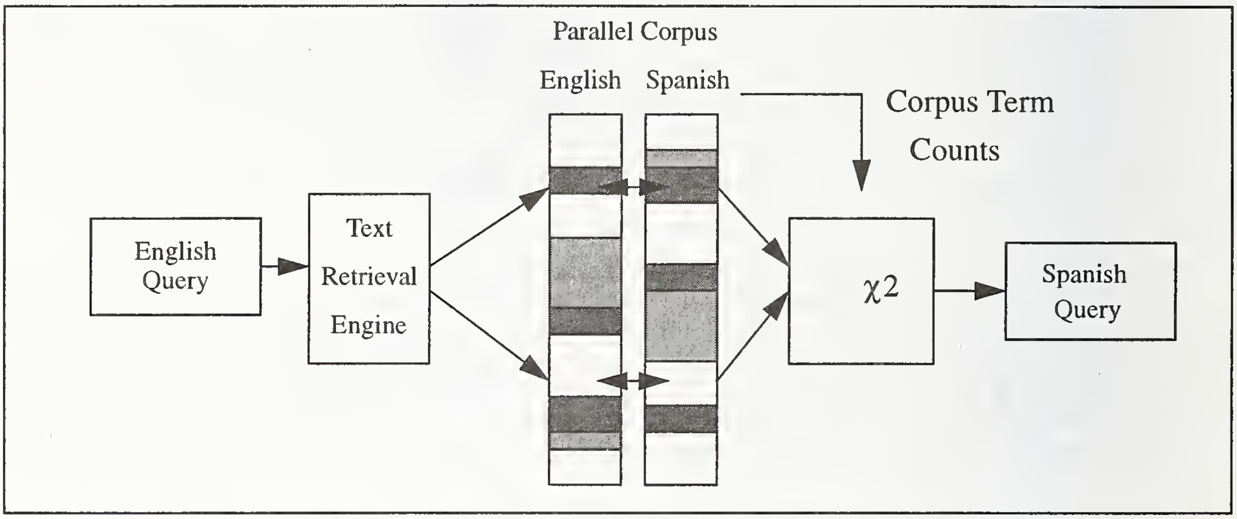


Figure 2. Extracting statistically-significant terms from parallel text look-up.

Evolutionary Optimization of Queries

If we could make a set of derived Spanish queries retrieve documents in a manner that is similar to the English queries over a training corpus, then the Spanish query could conceivably produce similar results on a novel corpus. One way to change Spanish queries is to add and remove terms. The number of possible unique deletions that can be performed on a 70 word query is quite large, however, making the direct examination of all possible modified queries effectively impossible.

We applied an evolutionary programming (EP) (Fogel, 1992) approach to modify a population of 50 queries. In an EP approach, an initial population of queries is needed along with a mutation strategy to modify queries. Optimization then proceeds by evaluating the comparative fitnesses of the queries, mutating a selected sub-population of the queries to produce “offspring” solutions and re-evaluating the queries iteratively until a suitable number of generations have passed. Our EP approach considered the comparative evaluation of document score vectors as an objective measure of the relative fitness of a query to the collection.

The initial queries for this test were the queries from the high-frequency lookup strategy discussed above. Previously, we have used a lexicon to generate initial queries (Davis and Dunning, 1995). The mutation strategy applied between one and ten modification operations to each of the 50 queries per generation and collected only the best 10% of the queries to propagate into the next generation. Optimization proceeded for 50 generations, resulting in a wide range of changes to each query. The fitness changes of the queries over 50 generations are shown in Figure 2.

The types of queries produced by this system typically showed the repetition of key terminology combined with the elimination of irrelevant terms. The fitness judgment for a query was based on comparative retrieval results using a training corpus of only 80,000 aligned sentences. The entire 680,000 aligned sentences were not used because we were uncertain that these results would be evaluated. We therefore intended an in-house evaluation over the remainder of the corpus should TREC evaluation be unavailable. Also, the retrieval engine for training was a traditional vector-based engine and not the same engine that was used to evaluate the TREC retrieval results (which was unavailable at the time of query preparation). Table 5, below, shows several of the resulting queries from the EP method.

Table 5: Evolutionary-Optimized Spanish Queries

Q#	Hand-translated English	Evolutionary Optimized Spanish Queries
26	Indicators of economic and business relations between Mexico and European contries.	Checoslovaquia En nacional Egipto Filipinas Portugal Finlandia gubernamentales Unidas una sesiones Mundial México resolución no un países organizaciones sus su República al sobre que en la Egipto nacional Filipinas Conferencia países México Checoslovaquia México México Egipto México México una Finlandia mujer México Egipto las se Finlandia Egipto como Comisión información E/CN sobre un Unidas General Unidas desarrollo países Finlandia Filipinas México actividades un nacional no Conferencia Filipinas Checoslovaquia Portugal nacionales Conferencia México República Egipto México al nacional proyecto México Secretario mujer que proyecto Filipinas que México Filipinas Finlandia la México En Checoslovaquia mexicana mexicano México
27	Indicators of economic and business relations between Mexico and African contries.	Egipto Los servicios Colombia Asamblea Naciones Unidos documento sus Argentina En General una al países Estados sobre un República con México del en una Colombia México servicios una México que Estados Egipto México en México siguientes Argentina trabajo Egipto México Asamblea documento Egipto Argentina República con de Secretario trabajo México principios la aplicación Colombia Argentina DE Egipto Colombia han las aplicación General Colombia Argentina servicios Colombia un documento han México los una en las México México con mexicana mexicano México
36	Indicators of the Mexican Navy's (naval and marine forces) strengths and weaknesses.	nivel respecto internacional internacionales Los lo marina sistema todos DE programas Asamblea General entre Consejo Mundial actividades países no climáticos Unidas como una por del para los en el la de climáticos marina marina R en sobre climáticos las marina Consejo los fracciones aplicación Consejo marina al período marina respecto marina entre zonas Mundial respecto marina zonas marina un sobre Gobierno climáticos todas sin sus mundial marina sus climáticos marina marina sus marina marina marina fracciones Estados del General marina por Los las En mexicana mexicano México
44	Information about Mexico's computer industry.	forma nuevas particular resultados contra económico fin lo medios Comité período sesiones Comisión internacionales Estados computadoras cuestiones mediante medio programa sistema computadora actividades El así servicios al sus países como su desarrollo una Naciones la La computadora organizaciones para actividades para computadora internacional así forma información computadora como período particular del computadoras el con Se computadoras General computadoras actividades programa computadora computadoras período del como computadora lo forma computadora computadora desarrollo ese servicios En computadora se sobre lo computadoras país que computadoras computadoras económico En computadora computadora computadoras mexicana mexicano México

Singular Value Decomposition and the Translation Matrix

The final query translation method was a radical departure from the others, but is derived from earlier work by Dunning and Davis (1993) and Landaurer and Littman (1990). This method is at heart a numerical approach to derive a translation matrix from parallel texts.

Let us suppose that there exists a translation operator that translates one query into another and, furthermore, that the queries can be represented as vectors of real-valued weights. This latter assumption is reasonable for virtually all text retrieval systems that treat queries as unordered "bags of words." In practice, a linearity assumption is reasonable in that translations of separate

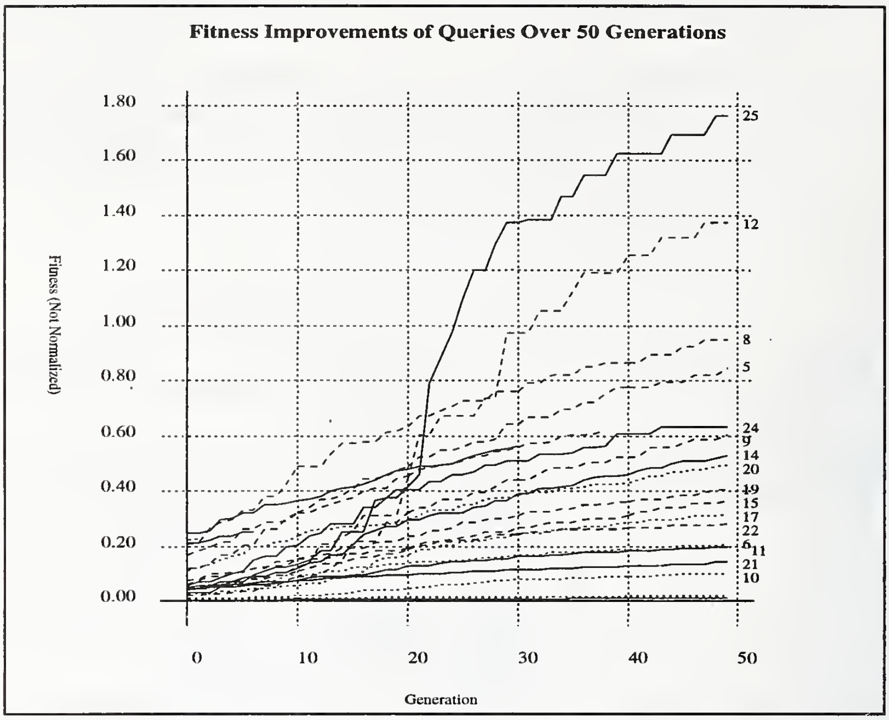


Figure 3. Fitness improvements of queries 1-25 over 50 generations during training. For TREC purposes, query 1 is equivalent to query 26, and so forth.

sections of texts are generally equivalent to translations of the text as a whole. This assumption breaks down below the level of the sentence, but is a useful approximation at the sentence level and above.

More specifically, given that we have a large number of short pieces of text which are translations of each other, we can use these short texts to form a set of linear equations which, when solved, yield a translation matrix T. Since T will be approximately a d by d matrix, it would seem that we would need millions of these translated bits of text. In fact, we can solve for an underdetermined least squares solution for T using singular value decomposition. This solution method avoids the numerical problem of not having enough samples, as well as handling the inevitable situation where the derived equations will not be quite consistent. The effect of using an underdetermined solution should be to preserve any ambiguity present in the query relative to the original parallel text.

In this effort, we applied a QR-decomposition technique to reduce the complexity of calculating the singular value decomposition, resulting in query translation that took only a matter of seconds on a SPARC 10. The generated queries are given in Table 6. It should be noted that these numerical methods are still very preliminary and relatively untested. One early result that we discovered was that English terms occurring in the Spanish text and vice-versa are highly singular features for the translation matrixes. These early trials must therefore be treated with caution.

Table 6: SVD generated queries

Q#	Hand-translated English	Corpus High-Frequency Spanish
26	Indicators of economic and business relations between Mexico and European contries.	Exteriores Relaciones Guillermo Bedregal Culto Ioan Bolivia Ministro documento párrafos México con parte reproducido oficiosas ex Simone decisión ° período Voicu Rumania externas Ayuda titulado si Gutiérrez asimismo decían mexicana mexicano México
27	Indicators of economic and business relations between Mexico and African contries.	costeras Los constituir INTERES MUNDIAL principales probablemente cambios bien curso profundamente posibles DE pobladas PROBLEMAS sí comprender particular contiguas Ministro próximo Las verán Culto donde pronosticado camino climáticos Zelandia causados mexicana mexicano México
28	Indicators of economic and business relations between Mexico and Asian countries, such as Japan, China and Korea.	informe aprobó octubre Junta sobre Voicu Mesa sesión Conferencias Finlandia período Guillermo básicos ° Consultivo UNCTAD/GATT Sucedáneos problems health Tungsteno relaciones México Ioan Arabes Exteriores grupos aportaciones credenciales Tal reuniones mexicana mexicano México
29	Indicators of economic and business relations between Mexico and Canada.	Exteriores Relaciones Guillermo Bedregal Culto México Bolivia Ministro Ley ideologías Rumania ejemplo cuerpo Otra resguarda provocarlos afectan étnico Voicu racistas exige aludida Tadanori Gutiérrez contribuirá cinematográficas mexicana mexicano México
30	Are there sports programs and exchanges between Mexico and the United States?	Exteriores Relaciones Guillermo Bedregal Culto Finlandia Bolivia Ministro relacionados programas Rumania sí serie conjunto distingue denominan Unión Soviéticas determinarse motivos México Voicu asociación convenios integrado Nam Gutiérrez del SIDA entre mexicana mexicano México
31	What measures has the Mexican government taken to resolve the quarrel with the rebel Zapatitas in the state of Chiapas?	mexicanas desequilibrios ecológicos Realiza arreglar costas presidente Pacífico evitar cuantía Mexicano recibido NASA información Comité siguiente El diga métodos futuro Nigeria junto Cirugía - Consejo propuestas archivado embargo comunidad actividades federal limitada mexicana mexicano México

RESULTS AND EVALUATION

The TREC evaluation procedure was an ideal approach to evaluating the effectiveness of the lexicon and corpus-based translation methodologies presented in this paper. Some caveats have to be considered with regard to the overall value of TREC results, however, including:

1. The different domains of text covered by the UN corpus and the target corpus for TREC-4 evaluation.
2. The English queries were hand-translated versions of the Spanish TREC queries that may not fully articulate the retrieval profile of the original Spanish query over the TREC corpus (a good test of this would be to translate back the English queries by hand with another translator and see whether the retrieval results differ.)
3. The unavailability of the *same* text retrieval engine for both training phases and for generating the final retrieval results.

Of all of these considerations, (3) is perhaps the most important. The Inquery Spanish retrieval engine was not fully functional until a few days prior to the submission date and therefore was unavailable for optimization in the case of EP evaluation iterations or for finding parallel text segments for query term extraction in all of the other cases. A straightforward idf-based retrieval engine was therefore substituted. What effect this has had on the ultimate results remains the subject for future research.

Despite these caveats, the full set queries were successfully run for each of the methods and results were evaluated at NIST. All of the methods resulted in substantial reductions in the precision-recall averages for the queries, although notable exceptions existed in which individual queries outperformed, or performed approximately as well as, the original Spanish queries. The former is a matter for some elaboration. It is always conceivable that a query modification process may result in better performance with respect to the original query but it is also very unusual for a process that set out to perform a radical transformation with the intent of "recreating" that query to actually outperform the original. This last result is a matter for further analysis.

Precision-Recall Averages

The average precision-recall curve for all 25 queries is shown in Figure 3, below.

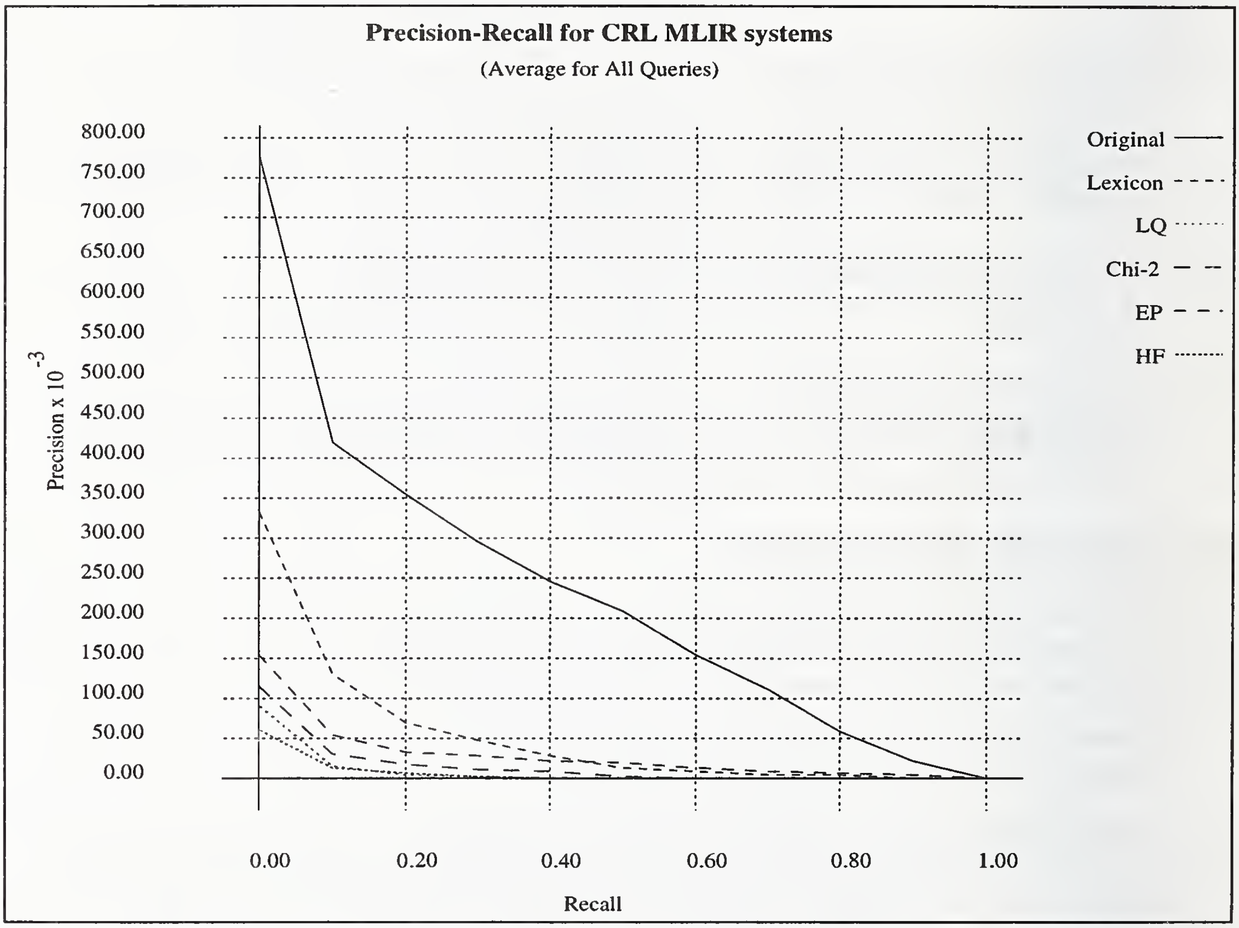


Figure 4. Average precision-recall curves for 25 Spanish queries.

The top line, representing the performance of the original Spanish query shows a competitive TREC text retrieval system. Below that are the lexical transfer curves, followed by the EP method, the Chi-squared approach and the SVD and high-frequency methods, respectively. Of special interest is the bow in the performance of the EP method in the high-end of the recall curve. In several cases (below) the EP method appeared to promote higher precision at higher recall rates than the other methods.

Some Interesting Anomalies

Although the overall results are somewhat discouraging, individual retrieval results show some startling anomalies. Figure 5 shows the precision-recall curve for queries 30, 36 and 44.

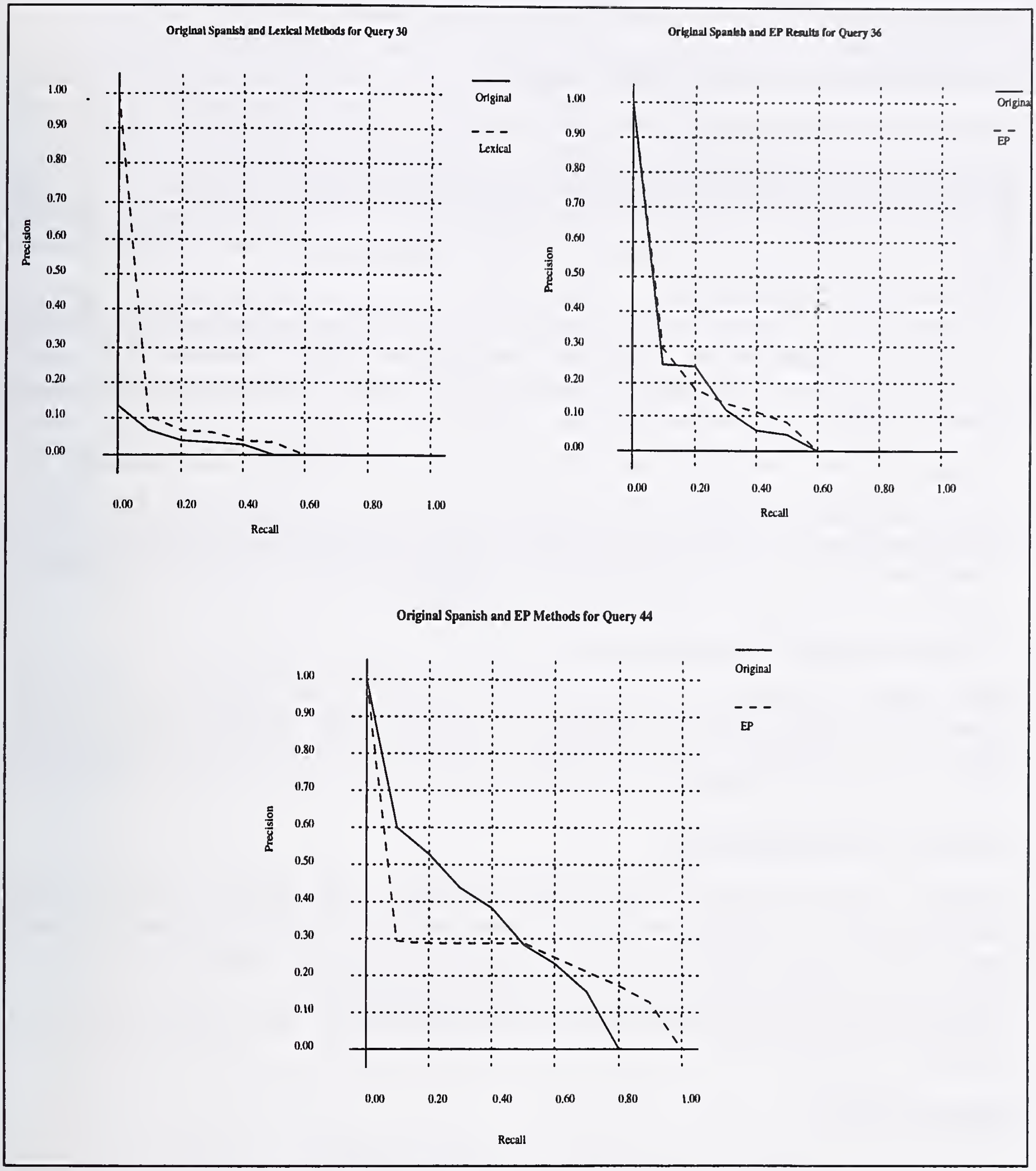


Figure 5. Anomalous results for Queries 30, 36 and 44.

The precision-recall curve for query 30 shows a situation where the lexical methods actually outperformed the original query substantially. The Inquery performance on the original query is probably suspect here; it may very well be that the Spanish morphology in Inquery failed to properly stem/expand the key term “deportivos”, while the lexical method generated “deporte.” Query

36 and 44 show the interesting property of the EP method to produce relatively high precision at higher rates of recall. Why this happens is unclear at the moment.

CONCLUSIONS AND FUTURE WORK

The five query translation methods proposed in this paper all utilize slightly different approaches to the problem of automatic query translation. The only approach not covered is a full machine-translation system. Although the overall results are not encouraging, the several anomalous results reported here do suggest that under certain, as yet to be determined, circumstances lexical and EP-based methods may outperform or perform as well as original queries. Future efforts will therefore focus on the following areas:

- Establishing an on-domain corpus of parallel text. In a setting where translators are constantly creating new translations, the availability of on-topic parallel texts is common. In our pursuit of a MLTR system for real-world applications, we therefore believe that the availability of on-domain corpora is a realistic assumption.
- Using the Inquery text retrieval engine for both query translation and results generation.
- Combining lexical and EP methods to gain the best properties of both systems. Early trials (Davis and Dunning, 1995) demonstrated that such a system appeared to work well when evaluation was only over a novel section of a corpus that was also used for training. Further evaluations are needed, however, at the level of TREC.
- Continued research on the SVD methods.

Finally, further examination of the TREC-4 results is needed. The results presented herein were only available five days prior to the deadline for this paper, limiting the depth of analysis that we could perform on the results. Closer examination of the results will likely suggest additional ways that the query translation methods can be improved.

ACKNOWLEDGEMENTS

The Inquery system runs were graciously donated by the University of Massachusetts at Amherst information retrieval research group. CRL is indebted to UMASS for taking the time to run and submit the numerous query result collections to NIST. We also wish to thank NIST for the Herculean effort to evaluate the entire collection of five runs of queries.

This research was funded under contract number MDA904-91-C-6153 from the United States Department of Defense.

REFERENCES

Davis, M. W., T. E. Dunning, and W. C. Ogden (1995) Text Alignment in the Real World: Improving Alignments of Noisy Translations Using Common Lexical Features, String Matching Strategies and N-Gram Comparisons. In *Proceedings of the Conference of the European Chapter of the Association of Computational Linguistics*. University College Dublin. March 1995.

Davis, M. W. and T.E. Dunning (1995) Query Translation Using Evolutionary Programming for Multi-Lingual Information Retrieval. In *Proceedings of the Fourth Annual Conference on Evolutionary Programming*, San Diego, Evolutionary Programming Society, 1995.

Dunning, T. E., and M. W. Davis (1993a), A Single Language Evaluation of a Multi-Lingual Text Retrieval System. In NIST Special Publication 500-207: *The First Text Retrieval Conference (TREC-1)*, ed. D.K. Harman, Computer Systems Laboratory, NIST.

Dunning, T. E., and M. W. Davis (1993b), Multi-Lingual Information Retrieval. *Memoranda in Computer and Cognitive Science*, MCCS-93-252, Computing Research Laboratory, New Mexico State University.

Dunning, T. E. (1993), Accurate Methods for the Statistics of Surprise and Coincidence, *Computational Linguistics*, 19, 1: 61-74.

Fogel, D. B. (1992), A Brief History of Simulated Evolution. In *Proc. of the First Annual Conference on Evolutionary Programming,* ed. D.B. Fogel and J.W. Atmar, 1-16. San Diego: Evolutionary Programming Society.

Landauer, T. K. and M. L. Littman (1990). Fully Automatic Cross-Language Document Retrieval Using Latent Semantic Indexing. In *Proceedings of the 6th Conference of UW Centre for the New Oxford English Dictionary and Text Research*, 31-38. Waterloo.

Salton, G. (1970). Automatic Processing of Foreign Language Documents. *Journal of the American Society for Information Sciences*, 21: 187-194.

Salton, G. and M. J. McGill (1983), *Introduction to Modern Information Retrieval*. New York: McGraw Hill.

Two Experiments on Retrieval With Corrupted Data and Clean Queries in the TREC4 Adhoc Task Environment: Data Fusion and Pattern Scanning.

Kwong Bor Ng and Paul B. Kantor
APLab, SCILS, Rutgers
The State University of New Jersey
4 Huntington St.
New Brunswick NJ 08903
Internet: {kng@zodiac, kantorp@cs}.rutgers.edu

Abstract

We report on several experiments in using data fusion to improve information retrieval, and in approximate text and 5-gram mathcing methods for retrieval of corrupted text, in the TREC context.

1. Data and Systems

We report on two experiments here. Our plan was to do all experiments on the whole set of TREC4 data, using an IR program called MG (version 1.0, Witten, Moffat and Bell, 1994) as our indexing and searching engine for the experiment. This plan was frustrated (twice) and the official TREC4 submission used a hand-built scanning retrieval technique for dealing with corrupted data.

The reasons for failure of the original approach are lie in the lack of sufficient processing space to manage the file of 5-grams (generated from the 2 Gigabtyes of data files) without reprogramming ; (2) the n-grams approach produced too many distinct terms for MG. Therefore, we decided to conduct both the experiments on only the data of the corruption track.

There are three sets of data for the corruption track: one set of clean data and two sets of corrupted data (10% and 20% corrupted). We conducted two experiments on parts of these data, using different indexing and searching systems. The first experiment was about data fusion. We used the clean set of data as the collection file and MG as our indexing and searching engine.

We originally intended to implement 5-gram searching by using an available very fast indexing and query system, MG (Witten, Moffat, Bell 1994).

The idea was:

Convert the text to overlapping 5-grams
Separate all 5 gram-s by NewLine markers
Submit the resulting text to the MG indexer
Transform queries in the same way
Run the queries against the index, using the MG query software.

At the time of these runs we had 9GB of hard disk storage available, partitioned into 1, 3 and 5GB. In general, the 5-grammed version of a text is between 5 and 6 times as long as the text. Since the MG index builder requires a single input file, we had to pipeline the result of the 5-gram expansion into the indexer. However, in its present form, the MG software must build a file of the entire text, for retrieval. Thus we still could not fit the result of the of 5-gram indexing process for the full TREC4 collection into our available storage.

We therefore switched to the Category B corruption track activity alone. This smaller file (less than 0.5 GB) can be managed within the resource constraints mentioned above.

The second experiment explored a mechanism for retrieving corrupted texts. There was no data fusion involved. We used a pure scan-match-and-count program to test the effectiveness of using "dot-5-grams" (see below) to retrieve corrupted documents.

2. Experiment 1. Data Fusion of the Outputs From The Clean Data

For the experiment on data fusion, we used two different methods to retrieve documents from the clean data. The first method was a ranking retrieval method with word frequency statistics and the cosine rule applied to document vector and query vector to estimate their similarity. We call this the "word-based approach". The second method employed the same strategy except that instead of using the words directly, we used overlapping 5-grams to represent the collection text and the query text (see below). We call this the "5-gram-based approach".

We used the program MG (version 1.0, running on Unix) as our indexing and searching engine. MG employs a csh script "mgbuild" that executes the appropriate sub-programs in the correct order to build a searchable database. We edited the mgbuild into two versions, one for the word based approach and the other for the n-gram-based approach.

2.1 Word Based Retrieval on the Category B Data. Adhoc Queries.

The data for the corrupt track include the WSJ and SJMN files from Disk 2 and Disk 3, in compressed form. We decompressed and concatenated them into one large text file (we called it the clear collection file, 475 Mbytes), then piped it to the first version of the modified mgbuild which accepts a text file piped in as input to construct the MG database. During indexing, each document is assigned an MG document number. We maintained a dictionary program external to MG to translate MG document numbers to original document numbers.

MG uses a sub-program "mgquery" to do the searching. We divided our queries expansion into two steps: preliminary and refined. In the first step, we used the 50 topics as queries and retrieved the top 50 documents (i.e., with highest cosine scores) for each topic. These could serve as expanded queries. We read the output to identify appropriate documents and used them as the refined expanded queries for the second round of search. For topics 223, 233, 245, and 246, we found no appropriate documents so the refined expanded queries were the same as the preliminary queries. For the other topics, we successfully identified some number of relevant documents (1 to 8) to serve as the refined expanded queries.

In the second step, we used a nawk script to pipe into mgquery the 50 refined expanded queries and, for each topic, get the MG document numbers and the relevance scores of the top 1000 documents output by MG. We call this output "clean words output".

2.2 N-gram Based Approach

2.2.1 Formation of 5-grams.

In this approach, we decomposed the collection texts and the query texts into their constituent n-grams and used the n-grams instead of the words to do the indexing and searching. Since n-grams of length 5 balance computational time and performance (Damashek, 1994), we used 5-grams. Before decomposing the texts to 5-grams, we translated all punctuation to period (.), and we escaped upper case letters to slash (\) followed by the corresponding lower case letter. All white space (tab, space, newline and the other non-alphanumeric codes) were translated to tilde (~). For example, before indexing or searching, the text:

```
Pear?
No. Orange!
```

is translated to

```
\pear.~\no.~\orange.~
```

The text is then decomposed into overlapping 5-grams. For simplicity each 5-gram is placed on a new line. (This wastes paper, but not disk space).

```
/pear
pear.
ear.~
ar.~/
r.~/n
.~/no
~/no.
/no.~
no.~/
o.~/o
.~/or
~/ora
/oran
orang
range
ange.
nge.~
```

2.2.2 Construction of the Database and Index

We piped the collection file to our second version of the modified mgbuild script, which contained our own programs (sed and C, by C. Basu) to do the translation as described above. The output is called the 5-grams database. The relationship between the MG document numbers of the 5-grams documents and the original document numbers in the collection file remains the same as in the word-based approach, and we use the same dictionary program for translation between them.

2.2.3 Retrieval

We wrote another nawk script to do the retrieval. It translated and decomposed the texts of the 50 topics into the required 5-gram format, piped them to mgquery, searched the 5-grams database, and for each topic found the MG document numbers and relevance scores of the top 1000 documents output by MG. We call the output "clean 5-grams output".

2.3 Data Fusion

The preceeding work results in two lists of the top 1000 documents, for each Topic in the Adhoc set (201-250). One list ("words") contains the top 100 documents resulting from the word based search, together with their relevance scores (calculated by a cosine method). The other contains the scores obtained when searching for the 5-grams of the query, in the 5-grams of the documents themselves. For each Topic the scores were normalized to lie in the interval [0,1], with the top ranked document assigned score 1, and the 1000th document assigned score 0. This maintains the relative size of gaps between documents.

We considered two schemes for combining the scores. One, called "mean" assigns to each document the mean of the scores which it has in the two lists. This approach is rigorously justified if (and only if) the odds that a document is relevant is a function only of this mean. This will occur if the odds are exponential in the mean, and if the distribution of documents in the space represented by the two scoring methods (words and 5-grams) has a simple product form [Kantor, 1995]. Methods of this type have been used with some success in earlier work by our own group and by others. [Belkin et al; 1993, Belkin, Kantor, Fox, Shaw, 1994]. The general notion of adding scores can also be supported by arguments about the scatter of retrieval methods about a true center, in a vector space approach [Kantor, 1994a].

The second scheme for data fusion implements an idea which can be stated easily in words [Kantor, 1994a], and which found some (very thin) support in our work in TREC3 [Kantor, 1994b]. The idea is: "if two schemes for retrieving documents are different, but each has some sense to it, then they are more likely to agree that relevant documents are relevant, than to agree that non-relevant documents are relevant." This has worked rather well with schemes which correspond to indiviudal human beings, when the notion of "agreement" was interpreted as overlap of Boolean retrieved sets. [Saracevic and Kantor]. In the present setting, with ranked sets, there are several possible interpretations of the Boolean overlap [Kantor, 1995; Belkin et al, 1994]. We chose to explore the variance of the rescaled normalized scores as such a measure.

Thus, from the two sets of ranked lists (words, 5-grams) for each topic, we produce two data fusion lists (mean, variance). Since not every item appears on both lists to be fused, we adopted the further rule that all items appearing on both lists are ranked above items appearing on only one list. All items on both lists were ranked in decreasing order of the mean score (for the mean fusion), and in increasing order of the variance of the two scores (for the variance fusion). These two official submissions for the uncorrupted data were labelled rutfum (rut=Rutgers; fu=fusion; m=mean) and rutfuv (v=variance).

3. Experiment 2. Corrupted Data

For the two sets of corrupted data (10% and 20% corrupted), at first, we planned to employ the same scheme as in Experiment 1. For example, to generate the "words" list we would use MG as the indexing and searching engine, with the same refined expanded queries as before. But the database would be built from the corrupted versions of the documents. However, the corrupted texts contain so many distinct words that we could not use MG to handle the indexing. We therefore abandoned, with regret, the fast indexing and searching capabilities of MG, and developed our own full scanning retrieval "engine". Scanning has been reported in the TREC environment previously. [Cavner; Mettler and Nordby]

Our scanning method is a modified version of the n-grams approach, motivated by the following considerations.

The process of translating and decomposing text to n-grams may generate at least n times as many terms (n-grams) for indexing. The "strangeness" of the spelling of the n-grams would also make ordinary stemming technique very difficult to apply. In addition, since the texts were corrupted, there were a lot of unusual words, and possibly some strange codes, that would make the inverted index file much larger than normal. In fact, we experienced some problems in attempting to use MG to construct the corrupted 5-grams databases. Since the purpose of this experiment is to test the mechanism, not the speed, we wrote our own scripts (csh, nawk, and perl) to do the searching and retrieval instead of using MG.

In corrupted texts, the spaces separating words might not be true correct word-boundaries as in normal text. In addition, any character of the text could be a corrupted space. This might cause a decrease in matching for the corrupted texts if we treated the spaces as word-separators and treated the non-space as non-word-separators. The use of dotting (described below) provides some measure of protection against this.

On the other hand, since the query texts were not corrupted, using spaces as word separators might compensate for the effect of the possible corruption of this function in the corrupted texts.

Our scanning process is enormously slower than the full indexing using MG. We were forced to remove all stop-words in the original topics before generating the 5-grams queries. This destroys semantic and syntactic relationships which would otherwise be parially represented by the overlapping 5-grams. With the queries reduced to bundles of non-stop-words, it does not make much sense to generate 5-grams across word-separators, and we did not. If the words had 5 characters or less we simply kept them untouched (for the sake of simplicity, in this paper we still call them 5-grams). For example, the 5-grams generated from the question:

> What is a pear and what is an orange?

would be:

pear
orang
range

3.1 Dotted 5-grams

After getting the 5-grams from the topics, we generated from them what we called dot-5-grams. That means, for each character in thc 5-grams, we generated the corresponding dot-5-gram by replacing that character with a dot. For example, the 5-grams "orang" would create 5 dot-5-gram, i.e., ".rang", "o.ang", "or.ng", "ora.g", and "oran.". Later in the process of matching, the dot would match with any character or code in the text.

The mechanism of the retrieval was simply the match and count mechanism, without weight consideration or vector calculation. We compared the dot-5-grams generated from each topic with the corrupted texts and counted the number of matches. Then we retrieved the top 1000 documents with the highest counts for each topic. These were then resorted in decreasing order of the ratio of the number of hits to the number of lines in the document (hits per line , or hpl). We call the outputs "10% corrupted dot-5-gram output" and "20% corrupted dot-5-gram output" respectively.

3.2 Dotted 5-gram retrieval

The pseudo-code for this dotted-5-gram retrieval is shown in Exhibit 1.

Exhibit 1. The dotted-5-gram algorithm

```
for each document
  hits=0
  for each line in the document
    for each 5-gram in the query
       for each dotting of the 5-gram
          if dotted-5-gram matches line
             hits=hits+1
          end if
        next dotting
       next 5-gram
     next line
   print document sequence number,
   print hits
next document
Sort documents by hits per line
```

The specific steps of this algorithm were implemented in nawk scripts, which run very slowly (typically 8 to 12 hours to complete a query with 25 words). To complete the analysis in time for this conference, the queries were reduced to content words alone, as described above.

3.3 Chance of missing a term

The rationale for this approach is illustrated by a calculation of the chance that a given term of length c will be missed by this algorithm, or will result in a specified number of hits h.

If there is no corruption, a term of length c will have h=1 hits if (c<6). If (c>5) then the number of hits will be equal to the number of overlapping 5-grams contained in the term, or h=c-4. For example, a term with 6 characters will have two overlapping 5-grams, both of which hit it.

In the presence of corruption at a level e, any of the characters has a probability e of being replaced by some other character. Disregarding, for the moment, the problem of false word boundaries, we see that the chance that all 5-characters in a particular position of the 5-gram window are intact is reduced to $(1\text{-}e)^5$.

The chance to have exactly h hits, if the term is of length c, is given by a more complex calculation, in which the number of times that each character is covered by a 5-gram is taken into account. For example, if the character in the middle of a nine character term is corrupted, then none of the 5-grams will hit. But if the first character is corrupted, then there will be four hits as the 5-gram window moves across the term. Details will be given elsewhere.

In sum, if a particular 5-gram is converted to all 5 possible dottings we improve robustness against corruption of the characters in the target term.

4. Performance Results and Discussion

4.1 Performance of Data Fusion by Mean Score. Clear Texts.

We look first at the results on the clear data. Here we used queries which were, in some cases, substantially expanded into a set of as many as 8 documents. The results are substantially better (we consider here only the mean fusion procedure). The results are not, however, at or above the median. The results for those topics for which we did not score below the median are summarized in Table 1.

Topic	OurScore	BestScore	Median
202	63	67	58
209	13	15	10
211	62	62	25
215	47	47	32
216	19	19	12
219	21	21	12
221	43	43	25
224	18	26	13
226	27	27	9
227	30	33	25
234	4	4	3
238	24	28	8
239	6	6	5
240	37	37	21
243	17	17	10
245	8	8	5
248	7	7	2
250	14	14	11

Table 1: best scores, our scores, and median of data fusion applied to clean texts. Results for the mean method of fusion are shown.

We are pleased to note that on quite a few of these our score is equal to the best score. But overall, we cannot say that the results of this scheme are "above the median". More precisely, with 95 percent confidence, they are below the median.

4.2 Performance of Dotted 5-grams and Scanning methods.

Although we have done them by different methods, we can compare our performance with that of the other systems, as reported by NIST. For 20% corruption the results are indeed dismal. Our results (measured in the number of relevant documents at 100 retrieved documents) was equal to the lowest of all reported scores in all but 7 of the 50 cases. Since there are only two entrants in this category, we have come in a poor second. Is there any reason for hope?

An optimist would point out that the n-gram and dotting techniques are expected to take advantage of having large queries. Thus we should not expect good results with sharply abbreviated queries, often stripped down to three or four words. In this sense, what we have done so far is simply establish that we can perform all the calculations called for, but not in a particularly efficient manner. Clearly a great deal remains to be learned. The specific topics on which our results were not the worst are 205, 208, 220, 224, 228, 229, 239 and 240.

5. Conclusions

5.1 Data Fusion: Results and interpretation.

The results of our data fusion experiments are summarized in Tables 1,2 and 3. We see that the results of fusion are not significantly better than the results of the two input schemes: term-based and 5-gram based. The results of fusion based on variance are too poor to be discussed. For the fusion based on mean scores we will try to understand why the fusion process did not score any better than shown. We will not enter into a case-by-case analysis. A preliminary examination of the scatter plots for the four cases in which (a) fusion was somewhat better than either of the input schemes and (b) there were more than a few relevant documents in the top 100 did not reveal any informative patterns.

Scheme	Avgp100	N (Topics)
fuv	0.050	49
fum	0.144	49
c20	0.041	49
c10	0.047	49
c0	0.050	49
word	0.149	49
5gram	0.138	49

Table 2. Average, over all Topics of the Precision at 100 documents.

Data Fusion Schemes and Input Schemes				
	fuv	fum	word	5gram
fuv	0	3	3	4
fum	39	0	19	17
word	39	10	0	15
5gram	39	10	19	0

Table 3. Comparison of two data fusion schemes with the performance of the two input schemes from which they are formed. The entry in each cell is the number of times the scheme labelling the row performed better than the scheme labelling the column. All comparisons are based on the number of judged relevant documents retrieved in the top 100 positions.

To show what kinds of patterns are sought, we plot (Figure 1) the scores for the top 100 documents, after mean fusion, showing the normalized scores based on normalization over the top 1000 documents. This permits us to show points representing documents for all of the topics on the same graph. The significance of scoring suing the term rule is that a vertical line sweeps across the graph from right to left, and documents are included as the line passes them. For the 5-gram rule the line sweeps down the plot, from the top, and documents are included as the line passes them. Finally, for the mean rule (which is logically equivalent to a sum of scores with equal weights) the line makes a slope of 45° with the axes. There are some interesting features of this plot to which we return below.

In Figure 2 we show a "zoomed" view of the sub-plot for documents both of whose normalized scores fall in the interval [0.2-0.5]. Examining Figure 2 we see that when all three schemes have just picked up the document which is represented by the point at the intersection of the three lines, the quadrant above and to the right of the point represents documents which have been included under all three schemes. The triangle on the lower right (a) includes points corresponding to documents which are retrieved under the f-gram scheme, and not under the mean fusion. On the other hand, the point in the trapezoid on the upper left (A) are included under the mean fusion scheme, but not included under the 5-gram scheme. Thus the mean fusion will be better than the 5-gram scheme if and only if regions of type A generally contain a higher ratio of relevant to not relevant documents than do the corresponding regions of type a. This relationship can be explored systematically. For the present we simply note that the graphic representation does not suggest, to the eye, any such preference for the mean fusion scheme.

Returning now to the plot of Figure 1, we consider the overall structure. The points are quite concentrated along the diagonal. This means that the two schemes (terms and 5-grams) are not providing independent information about the retrievable documents. This is precisely the situation in which data fusion will not be expect to work. In the limit, if the points lie on a single line (even a curved line) there is no advantage to knowing both coordinates of a point, as either determines its position in the overall ranked set.

For the most part, the points far from the principal diagonal are dots ("."), representing non-relevant documents. This is to be expected,a nd is summed up by the proposition that "good retrieval schemes are more likely to disagree about the score assigned to non-relevant documents than about the score assigned to relevant documents". This is the motivation, as noted above, for the fusion scheme based on variance. However, for the two schemes combined here, the fusion based on variance does extremely poorly. This is precisely because nearly all the documents, whether or not they are relevant are clustered close to the principal diagonal (which corresponds to variance 0).

We finally note a peculiar feature of Figure 1, which is the region in the lower right which is far from the central diagonal, but is occupied entirely by relevant documents. This is exactly the kind of phenomenon postulated when data fusion is viewed as a form of adaptive pattern recognition in the space of multiple attributes. Of course this region is far too sparsely occupied for us to apply that method here, but it serves to illustrate the concept. Mathematically this region is defined by high values of the difference *(x-y)*, where *x* represents the term score of the document. In fact, selecting the cases for which *(x-y)> 0.4* we find that 10 of these cases come from topic 223, and ten of them come from topic 225. The significance of this is not yet understood. It appears that for these topics the difference of scores might be an effective rule of combination. But this conclusion may be the equivalent of statistical data mining, and not significant.

5.2 Scanning and Confusion

Turning now to the retrieval on corrupted data, we show, in Tables 1 and 3 the results for retrieval using the scanning scheme on uncorrupted, 10% corrupted, and 20% corrupted text. These schemes degrade as expected. However, they perform poorly in comparison to schemes which index all the 5-grams, and use the original texts as queries. Recall that our scanning scheme uses severely truncated queries in order to control processing time.

Corrupted Data. Scanning schemes			
	c20	c10	c0
c20	0	5	5
c10	17	0	7
c0	26	17	0

Table 4. Comparison of performance of the scanning scheme on data at different levels of corruption. The entry in each cell is the number of cases in which the scheme of the row scored better than the scheme of the column.

5.3 Overall Conclusions

Discussion and Prospects.

We find two principal results. First, the 5-gram indexing scheme and the term-based indexing scheme (which includes a stop list) seem to provide scores for each document which are too similar to support a priori fusion schemes based on the variance, the mean, or a weighted sum. While we present here only the scatter plots for all 49 topics together, the conclusion is much the same for the individual topics as well. Second, the weak-match scanning scheme used with very short queries does not handle corruption as well as the several indexing schemes reported by other participants [Buckley et al; Frieder et al; Huffman ; this volume]. Experiments are planned to explore the dependence of this performance on three possible improvements: (1) use of an "inverse document frequency" weight to decrease the importance of patterns which are matched frequently (2) the use of a center of gravity transform as an alternate method for dealing with the issue of document frequency (3) extension to longer queries, based on relevant retrieved documents.

It is clear that the two methods reported here (data fusion by mean score, for the clear texts, using refined expanded queries composed of zero to eight relevant documents; and scanning with key-word queries and dotted 5-grams) do not advance the state of the art, as represented (at a minimum) by the current TREC entrants. Without regard to the state of the art, fusion by minimum variance of the normalized score performs dramatically more poorly than fusion by the mean. For unfunded research, and newly developed systems, we feel that these results are not embarassing. In particular, a number of possibilities are open for further exploration. They include:

1. Revisiting the original (word and 5-gram) lists which entered into the data fusion. Do either of these lists perform significantly better than fusion by the mean, which is our best official entrant.

2. Examining alternative fusion schemes. The symmetric sum, represented by the mean, and the variance, represent only two among a continuum of possible rules for data fusion.

3. Improving the completeness of the treatment of corrupted texts. Since results, in comparison to the median, are much better for clear texts than for corrupted texts, we might, as noted above do better on the corrupted texts by expanding the queries and/or doing some variant of data fusion.

Finally, the MG program which we applied to the clear text supports important weighting and stemming features whose analogues for the corrupted case are not apparent. In general, our design philosophy is to avoid language-specific devices (morphologies, externally generated thesauri, etc.) but there may be some analogues to the corpus-generated thesaurus which are appropriate for the case of corrupted data.

6. The Authors

K.B. Ng is a graduate student in the PhD program in Communication, Information and Library Studies at Rutgers, and holds a teaching assistantship there. P. B. Kantor is Professor in the School of Communication, Information and Library Studies at Rutgers, where he directs the APLab, and is also a Member of the Center for Operations Research, and a Faculty Associate of the Project in High Performance Computing and Design.

7. Acknowledgements

We are, in the words of T. Williams, dependent upon the kindness of others again this year. First to the team in Melbourne, for creating and making available the MG engine, and for providing support at odd hours of the day and night as we tried to bend it to tasks for which it was never intended. Particularly to Tim Sinnott, technical support. At Rutgers, Chumki Basu, a graduate student in computer science wrote several filters in C, and also scanned the great majority of the retrieved documents in order to build the refined enhanced queries. Oleg Goldenstein, a graduate student in Library and Information Studies joined the project late, and worked on a compiled version of the scanner, which has not yet been implemented. As always, intra-departmental rivalry with the team headed by N. Belkin, of Library and Information Studies, working on relevance feedback and interaction was a constant and pleasurable stimulus. The support of Dean Richard Budd for the Alexandria Project Laboratory (APLab) has been essential in the development of the necessary infrastructure. The APLab is also supported by grants and contracts from private and federal institutions.

8. References:

Damashek, M. (1994) Gauging Similarity via N-Grams: Language-Independent Sorting, Categorization, and Retrieval of Text, Science, 267, (10 Feb., 1995) 843-848.

Belkin, Kantor, Fox and Shaw. Combining Evidence for Information Retrieval. in D. Harman, Ed. Proceedings of the 2nd annual Text Retrieval Conference. NIST 1994. pp35-44

Belkin, Kantor, Cool and Quatrain. (1995). Combining the Evidence of Multiple Query Representations for Information Retrieval. Information Processing and Management 31(3)431-448.

Cavnar, William B. N-Gram-Based Text Filtering for TREC-2. D. Harman, Ed. Proceedings of the 2nd annual Text Retrieval Conference. NIST 1994. pp171-179

Kantor, Paul B. Data Fusion in Information Retrieval: Towards a Theoretical Foundation A: Vector Simulation Models.. APLab Technical Report. APLab/TR-93/3. Rev. Sept. 1994a.

Kantor, Paul B. Decision Level Data Fusion for Routing of Documents inthe TREC3 Context: A Best Case Analysis of Worst Case Results. in D. Harman, Ed. Proceedings of the 3nd annual Text Retrieval Conference. NIST 1995. pp

Kantor, Paul B. Tutorial on Data Fusion in Information Retreival (1995). ACM SIGIR.

Mettler, Matt, Nordby, Fritz. TREC-II Routing Experiments with the TRW/Paracel Fast Data Finder. in D. Harman, Ed. Proceedings of the 2nd annual Text Retrieval Conference. NIST 1995. pp319-332..

Saracevic, T, Kantor, PB, A Study of Information Seeking and Retrieving. III Searchers, Searches, Overlap. JASIS.39:178-194 (1988)

Witten, I. H., Moffat, A., and Bell, T. C. (1994) Managing Gigabytes. New York: Van Nostrand Reinhold.

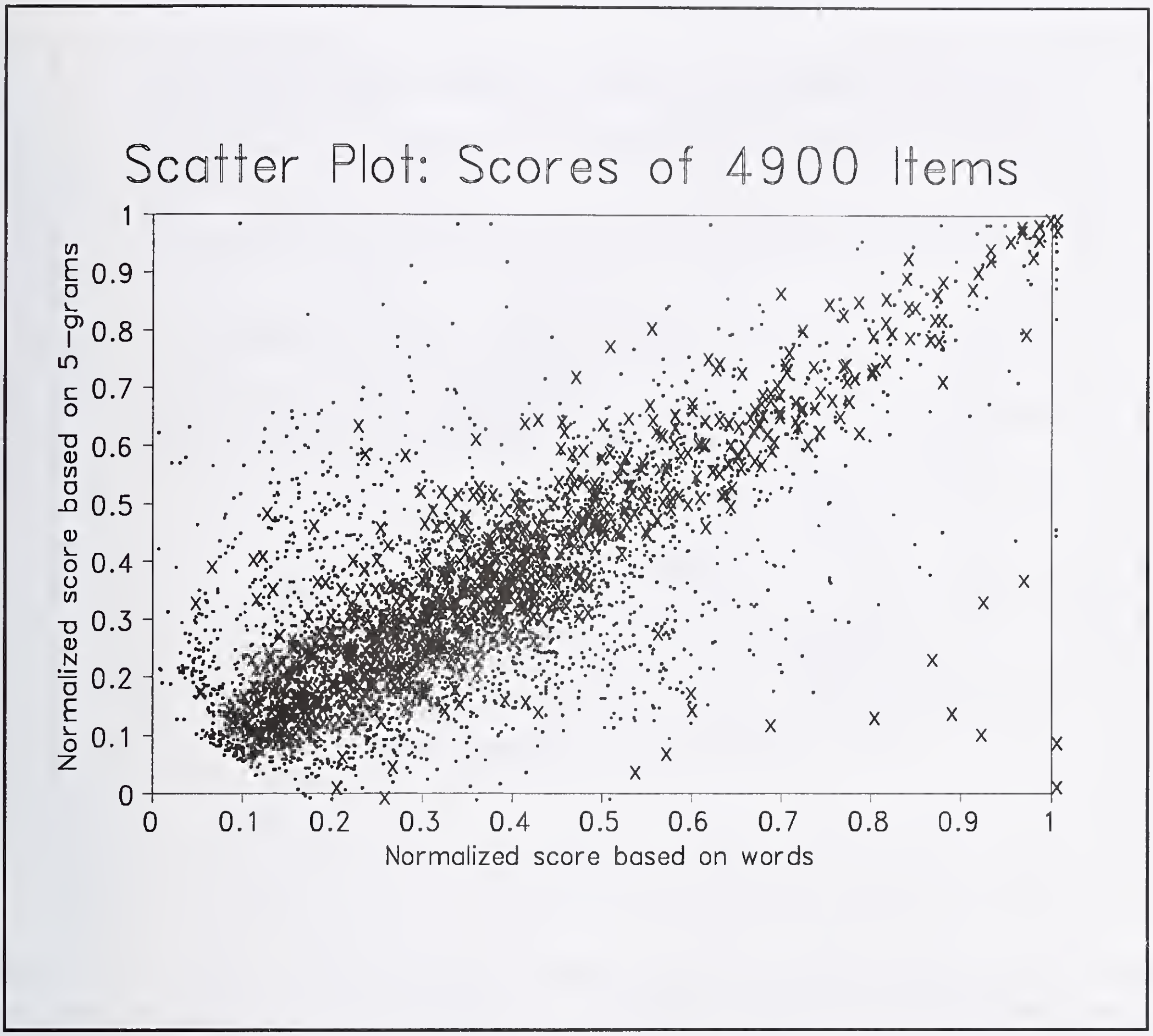


Figure 1. Scatter plot of the scores assigned to the top 100 documents for each of 49 adhoc topics, "x" represents a relevant document and "." represents a non-relevant document.

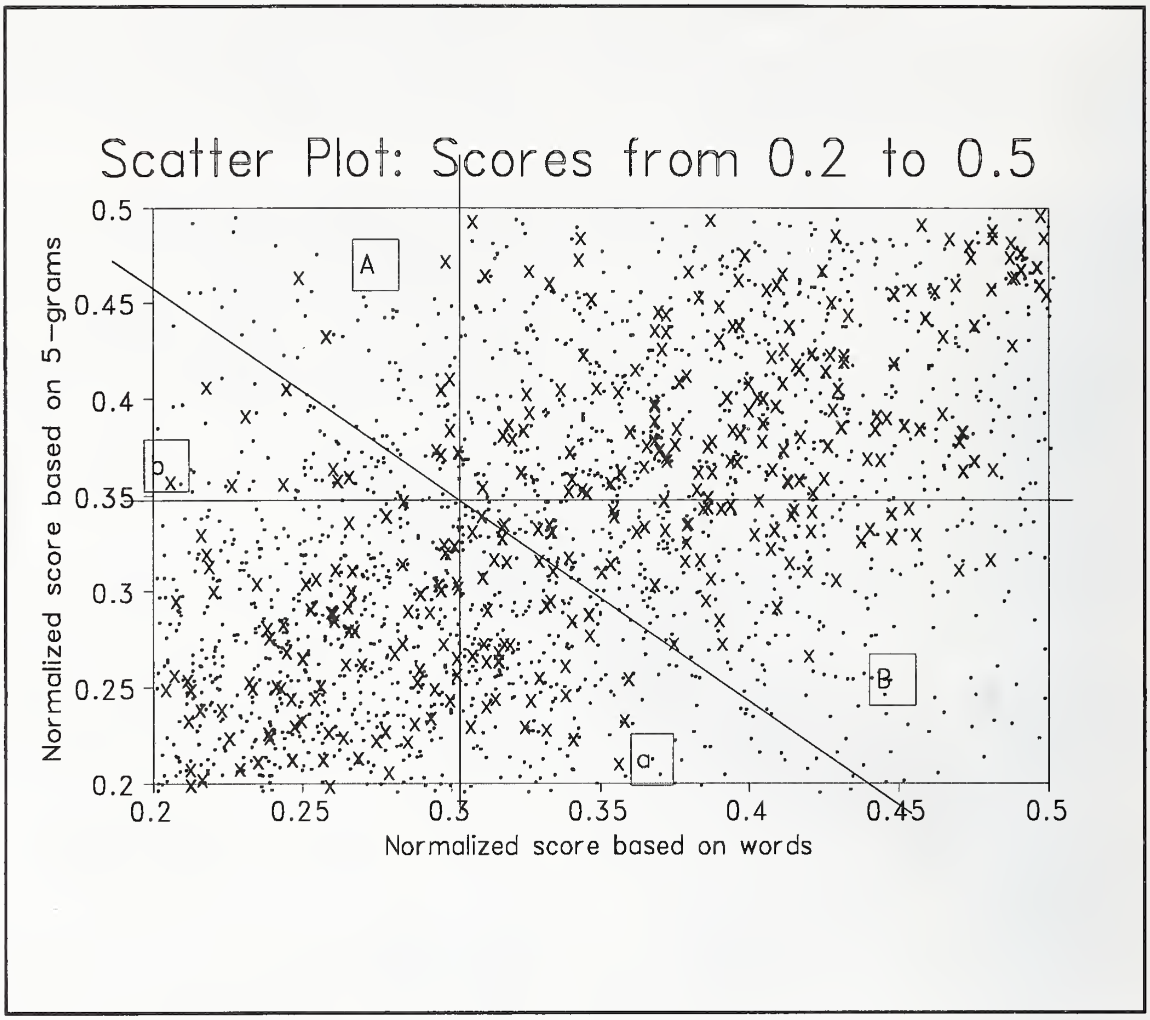


Figure 2. Enlarged portion of the scatter plot, showing level curves for term-based (words), 5-grams and mean. Graphic errors have changed to diagonal line, which should pass through the point 0.45 on each axis.

The Troubles with Using a Logical Model of IR on a Large Collection of Documents

Experimenting Retrieval by Logical Imaging on TREC

F. Crestani†, I. Ruthven*‡, M. Sanderson‡, C.J. van Rijsbergen‡

†Dip. di Elettronica e Informatica - Universitá di Padova - Italy
‡Dept. of Computing Science - University of Glasgow - Scotland

Abstract

The evaluation of an implication by Imaging is a logical technique developed in the framework of Conditional Logics. In 1993 a logical model of IR called "Retrieval by Logical Imaging" was proposed by some of the authors of this paper and tested using some classical IR test collections.

In this paper we report on the challenges posed by trying to apply such a model to a large test collection of the size of TREC-B. The problems we found and the way we put together ideas and efforts to solve them are indicative of the troubles one might find in trying to implement and experiment with a "complex" logical model of IR. We believe our efforts could set an example for other researchers working on logical models of IR to try to implement their models in such a way that they can cope with the size of real life collections, though preserving the formal "beauty" of their logical models.

*Address to which correspondence should be sent: Ian Ruthven, Department of Computing Science, University of Glasgow, Glasgow G12 8QQ, Scotland, UK; email: trec@dcs.gla.ac.uk

1 The use of non-classical logic in Information Retrieval

In recent years there have been several attempts to define *a logic for IR*. The earliest approaches were directed to the use of classical logic, like Boolean logic [Sal68], with few notable exceptions [Hil65]. The basis of a logical model for IR is the assumption that queries and documents can be represented by logical formulas. In order to retrieve a document an IR system has to infer the query from formulas representing the document. This logical interpretation of query and documents emphasises that *IR is an inference process* by which we can infer if a document is relevant to the query using information present in the document itself together with user knowledge. In classical logic inference is often associated with *logical implication*: a document is relevant to a query if it implies the query, that is if $d \rightarrow q$ is true. In IR it was soon realised that it is necessary to take into consideration the uncertainty inherent in this implication. A collection of documents cannot be considered as a consistent set of statements, and documents in the collection could even contradict each other. In order to cope with uncertainty a logic for probabilistic inference was introduced. If $d \rightarrow q$ is uncertain, then we can measure its degree of uncertainty by $P(d \rightarrow q)$. An early suggestion was to estimate $P(d \rightarrow q)$ by $P(q/d)$. However, the limitation of this approach explained by the triviality results of Lewis [Lew81] excluded that conditional probabilities could be used as probabilistic logic dealing with conditionals.

In 1986 Van Rijsbergen [vR86] proposed the use of a non-classical conditional logic for IR. The proposal initiated a new line of research that was followed by many researchers (see for example [Nie88, Nie89, CC92, Bru93]).

A few years later Van Rijsbergen proposed to estimate the probability of the conditional by a process called Imaging [vR89]. This idea was finally put into an implementation in 1994 when Crestani and Van Rijsbergen [CvR95] proposed a retrieval technique called *Retrieval by Logical Imaging* (RbLI). This technique enables the evaluation of $P(d \rightarrow q)$ and $P(q \rightarrow d)$ by Imaging according to a "Possible Worlds" semantics where a term is considered as a possible world. This technique exploits term-term relationships in retrieval by means of an accessibility relation between worlds based on their Expected Mutual Information Measure (EMIM).

This paper reports on the problems, solutions, and current results of the experimentation of Retrieval by Logical Imaging using a large collection of documents. The paper is structured as follows. In Section 2 we present the model, while in Section 3 we lay out the experimental settings for the implementation of the model. This is where the problems start. Experimenting with a large collection, of the size of TREC-B, poses considerable difficulties that are reported in Section 4. Section 5 describes our attempted solutions towards an implementation of the RbLI model that could cope with the size of the test collection. Section 6 reports our current results in the context of the TREC-4 initiative, "ad hoc" track. Further directions of investigation are described in Section 7.

2 Retrieval by Logical Imaging

Logical Imaging (LI) is a logical technique that enables the evaluation of a conditional sentence without explicitly defining the operator "$\rightarrow$" [Sta81]. LI is based on the Possible World Semantics [Kri71], that is on a semantics where the truth value of a logical sentence is evaluated in the context of a "world", or as Hughes called it in the context of a "conceivable or envisageable state of affairs" ([HC68], p. 75). According to this semantics the truth value of the conditional $y \rightarrow x$ in a world w is equivalent to the truth value of the consequent x in the closest world w_y to w where the antecedent y is true.

LI was extended to the case where there is a probability distribution on the worlds by Lewis [Lew81]. In this case the evaluation of $P(y \rightarrow x)$ causes a shift of the original probability P from a world w to the closest world w_y where y is true. Probability is neither created nor destroyed, it is simply moved from a "not-y-world" to a "y-world" to derive a new probability distribution P_y. This process is called "deriving P_y from P by imaging on y".

We will not go here into a detailed explanation of the Imaging process. The interested reader can look at papers by Stalnaker [Sta81], Lewis [Lew81], and Gärdenfors [G̈88] for more details.

We use Imaging in IR with the purpose of estimating the probability of relevance of a document by means of the probability of the conditional $d \rightarrow q$, by assuming that:

$$P(R \mid q, d) \approx P(d \rightarrow q)$$

We call *Retrieval by Logical Imaging* (RbLI) a model that produces a ranking of every document d_i in the collection based on an estimate of $P(d_i \rightarrow q)$. A detailed explanation of this model can be found in [CvR95]. Briefly, in RbLI we derive P_d by imaging on d considering all the index terms t in T as possible worlds. If so, $P(d \rightarrow q)$ can be evaluated as:

$$\begin{aligned} P(d \rightarrow q) &= P_d(q) \\ &= \textstyle\sum_t P_d(t)\ I(t, q) \\ &= \textstyle\sum_t P(t)\ I(t_d, q) \end{aligned}$$

where $P(t)$ and $P_d(t)$ are respectively the "prior" and the "posterior" probability assigned on the space T, t_d is the closest term to t for which d is true, or in other words, the most similar term to t that occurs in the document d, and $I(t_d, q)$ is defined as:

$$I(t_d, q) = \begin{cases} 1 & \text{if } t_d \in q \\ 0 & \text{otherwise} \end{cases}$$

The application of the above technique to IR requires a probability distribution on T so that for every t we can have $P(t)$, and a measure of similarity over the term space T to enable the identification of t_d. We will tackle this problem in Section 3.

LI provides the minimal revision of the "prior" probability in the sense that it involves no gratuitous movement of probability from world to dissimilar worlds. In fact, the revision of the "prior" probability necessary to make d certain is obtained by adopting the least drastic change in the probability space. This is achieved by transferring probabilities from each term not occuring in the document d to its closest (the most similar) term occurring in it, so that the total amount of the distance covered in the transfer is minimal. A detailed comparison between conditional probability and the conditionalisation performed by Imaging can be found in [Cro94].

For a practical example of the evaluation of RbLI let us suppose we have a document d represented by terms t_1, t_5, and t_6 and a query q represented by

t	$P(t)$	$I(t,d)$	t_d	$P_d(t)$	$I(t,q)$	$P_d(t) \cdot I(t,q)$
1	0.2	1	1	0.3	1	0.3
2	0.1	0	1	0	0	0
3	0.05	0	5	0	0	0
4	0.2	0	5	0	1	0
5	0.3	1	5	0.55	0	0
6	0.15	1	6	0.15	1	0.15
$\sum_t$	1.0			1.0		0.45

Table 1: Evaluation of $P(d \to q)$ by imaging on d

t_1, t_4, and t_6. Each of these terms has a "prior" probability associated with it, this is indicated by $P(t)$. Table 1 reports the evaluation of $P(d \to q)$ by imaging on d. The evaluation process is the following:

1. Identify the terms occurring in the document d (third column of the table).
2. Determine for each term in T the t_d, i.e. the most similar term to t for which $I(t,d) = 1$. This is done using a similarity measure on the term space (fourth column).
3. Evaluate $P_d(t)$ by transferring the probabilities from terms not occurring in the document to terms occurring in it (fifth column).
4. Evaluate $I(t,q)$ for each term, i.e. identify the terms occurring in the query (sixth column).
5. Evaluate $P_d(t) \cdot I(t,q)$ for all terms (seventh column) and evaluate $P_d(q)$ by summation (bottom of seventh column).

A graphical interpretation of this process is depicted in Figure 1.

3 Implementing Retrieval by Logical Imaging

As it has been introduced in the previous section, in order to implement the RbLI model we require:

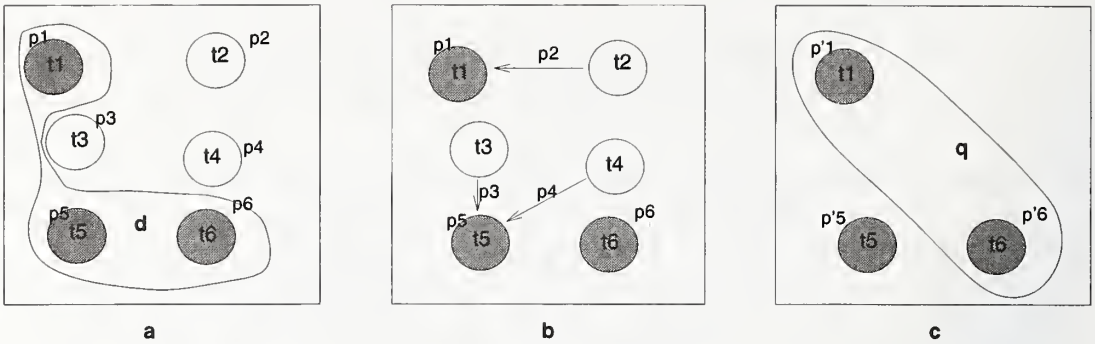


Figure 1: Graphical interpretation of the evaluation of $P(d \rightarrow q)$ by imaging on d.

1. a "prior" probability distribution over the index term space that should reflect the importance of each index term in the term space;
2. a measure of similarity (or alternatively a distance) between index terms;

These two requirements reflects the use of a Possible World Semantics, since they correspond to the probability distribution, and to the accessibility relation measure among the possible worlds [Lew81].

The problem of determining an appropriate "prior" probability distribution over the set of terms used to index a document collection is one of the oldest problems in IR and many models have been proposed for this purpose. The problem could be translated into finding a so called "measure of the importance of the term in the term space". In IR several discrimination measures have been proposed (see for example [vR79, RS76]) and there it is not clear which one should be prefered to the others. For the experiments performed in TREC we used the *Inverse Document Frequency* (*idf*) defined as:

$$idf(t) = -log \ \frac{n}{N}$$

where n is the number of documents in which the term t occurs, and N is the total number of documents in the collection.

Strictly speaking, this is not a probability measure since $\sum_t idf(t) \neq 1$, however since we assume it to be monotone to $P(t)$, we can use it instead of a proper probability function because we are only interested in a ranking of the documents of the collection, not in the exact probability values.

The problem of measuring the similarity between index terms in order to define a measure of accessibility among worlds is more difficult. It is very important to chose the appropriate measure since much of the power of RbLI depends on it. For our TREC experiments we used the *Expected Mutual Information Measure* (EMIM). The EMIM between two index terms is often interpreted as a measure of the statistical information contained in the first term about the other one (or vice versa, it being a symmetric measure). EMIM is defined as follows:

$$I(i,j) = \sum_{t_i,t_j} P(t_i,t_j)\ log\frac{P(t_i,t_j)}{P(t_i)P(t_j)}$$

where i and j are binary variables representing terms.

When we apply this measure to binary variables we can estimate EMIM between two terms using the technique proposed in [vR77] p. 130. Using this measure we can evaluate for every term a ranking of all the other terms according to their decreasing level of similarity with it. We store this information in a file which is used at run-time to determine for an index term the most similar other index term that occurs in the document under consideration.

4 Experimenting using a large document collection

In [CvR95] the performance of the RbLI model was tested using a the *Cranfield* document collection. In [Cv95] the RbLI model was generalised into the RbGLI model and both these models were tested using the Cranfield, the *CACM*, and the *NPL* document collections. In this last paper RbLI and RbGLI were compared with two prototypical classical retrieval models and the following experimental result was found ([Cv95], p. 298):

> " . . . we believe we have shown that *in principle* a probability transfer that takes into account a measure of similarity between the donor and the recipient is more effective in the context of IR than a probability transfer that does not take that into account. Most current probabilistic retrieval models are based on a probability kinematics that does not take into account similarity between terms or between documents, unless "ad hoc" weighting schemas, mostly based on clustering, are used. We would therefore like to suggest a further investigation into more complex and optimised models for probabilistic retrieval, where probability kinematics follows a non-classical approach."

In order to ensure that this result does not depend on the small size of the document collections we used, we decided to test these models on a collection of much larger size. Moreover, we decided to compare their performance with that of real IR systems, ones that could be recognised as a tough "benchmark".

We decided to procede in two steps: first implement RbLI and test it, and only later implement RbGLI. The RbGLI model is more computationaly demanding due to the introduction of a "probability transfer function" that enables the probability to be tranferred not only to a single t_d but to a set of them according to their respective distance to the term under consideration.

The implementation and testing of RbLI in [CvR95] and [Cv95] were quite heavy due to inefficient implementations of the probability transfer and to the complexity of the models. These problems, that were easily solved on small document collections, are much more difficult to tackle with a large document collection. In the following section we report on the challenge posed by trying to make the RbLI work on a large document collection. Our participation in TREC-4 was in the smaller part B collection which consists of $165,000$ documents.

5 Getting RbLI to work

Because of the anticipated high computational load of performing RbLI on the TREC-B collection, it was decided to initially run experiments on a sub set of TREC-B so as to prototype the RbLI software being developed for these experiments. Rather than remove documents from the collection, it was decided to reduce all documents (which are in fact news articles) to just their lead paragraph, which generally for news articles is a summary of the article. This reduction resulted in a 70Mb document collection. All work reported in this section is based on this modified collection.

The implementations of RbLI reported in Crestani and van Rijsbergen [Cv95] were performed on small test collections. Because of their size, it was possible to compute in a reasonable time the probability transfer of all terms in each document in the collection. On the TREC-B collection however, it was calculated that to perform this complete transfer would take too long given current computing resources. So methods of optimising the probability transfer were investigated.

5.1 Reducing the number of transfers

The first area looked at was the accessibility relation used to determine how probabilities are transferred, namely the EMIM measure. One of the features of EMIM is its ability to compute the relatedness between any two terms even if those terms don't co-occur, which means that in the case of RbLI it would be possible to compute probability transfers between all terms. The EMIM measure calculated between terms that don't co-occur however was found to be close to a small constant value, so for the experiments reported in this section a decision was made to restrict the EMIM calculations to only those terms that co-occur. When transferring probabilities onto a document's terms, any term that doesn't co-occur with that document's terms will have its probabilities uniformly distributed to all those document terms, so as not to lose its probability.

By calculating the EMIM between only co-occurring terms, the total number of calculations to be performed is reduced by around 95%. But it was felt that this reduction could and should be improved with further optimisations.

5.2 More speed

Now that probability transfers were reduced to just those terms that co-occur, the speed of RbLI on a document collection becomes proportional to the number of term co-occurrences in that collection. Therefore if we want to speed up RbLI, we need to reduce the number of these co-occurrences. There are a several ways in which this might be done. For example one could choose to only use those term co-occurrences where the two co-occurring terms appear in the same paragraph. Indeed this possibility might be exploited in the future, for the time being however it was decided to investigate the speed increase on RbLI when whole terms are removed from the collection. This technique was already proposed and tested by Crestani and Van Rijsbergen in [CvR95], where an intuitive explanation of its usefulness was given. We wanted to test its effectiveness on a large document collection.

In choosing terms to be removed, the question arises which type of terms should we concentrate on: (a) the few terms that occur in many documents, or (b) the many terms that occur in a few documents, indeed is there any difference between the two? To answer this question, we need to examine the imaging process in more detail.

The part of RbLI that is the most computationally intensive is the final stage where probabilities are transferred onto the terms of each collection document. The time taken to complete this stage is proportional to the total number of probability transfers that will potentially be made. The term "potential" is used because not all transfers will happen. RbLI demands that even if a term co-occurs with several document terms, that term will only transfer its probability to just one of those document terms, its most similar. Nevertheless each one of these potential transfers has to be considered by the RbLI software, so each potential transfer does add to the time taken to complete this task. The number of potential transfers can be calculated using the following formula:

$$number\ of\ transfers = \sum_{d=1}^{D} \sum_{t=1}^{T_d} O_t$$

where: D is the set of all documents in the collection, T_d is the set of terms contained in document d, and O_t is the number of terms that co-occur with t.

The formula for O_t is as follows:

$$O_t = \sum_{d=1}^{D_t} (N_d - 1)$$

where: D_t is the set of documents containing term t, and N_d is the number of distinct terms in document d

So, for example, given the choice of, case A, removing 1 term that occurs in 100 documents or, case B, removing 50 terms that each occur in 2 documents, we can use these formula to calculate which of these choices will reduce the number of transfers the most. For example, if we assume that each document contains 10 distinct terms, then the reduction in the number of transfers resulting from the two term removal cases is as follows[1]:

For the first case

$$number\ of\ tranfers(A) = 100 * 1 * (100 * 9) = 90,000$$

For the second case

$$number\ of\ tranfers(B) = 50 * (2 * *1 * (2 * 9)) = 1,800$$

From this, we conclude that efforts should be concentrated on reducing the small number of terms that occur frequently in the collection.

5.3 Reducing the small number of terms that occur frequently in the collection

In initial experiments a standard stop list (taken from Van Rijsbergen [vR79] p. 18) was used when indexing TREC, but in the light of the work described above, it was decided to investigate how retrieval performance would be affected when a bigger stop list was used.

Using a standard $tf \cdot idf$ retrieval system, the effect on retrieval performance from using a number of different stop lists was tested. The definition of a term's membership for these stop lists was based on the number of documents that term occurred in. Stop lists were generated for terms that occurred in more than 90% of documents, more than 80% of documents,

[1] The removal of term(s) has another minor influence on the time taken to complete RbLI: all terms co-occuring with the term(s) being removed will have fewer probability transfers to them. The effect of this influence however is the same for both cases, and so it need not be considered.

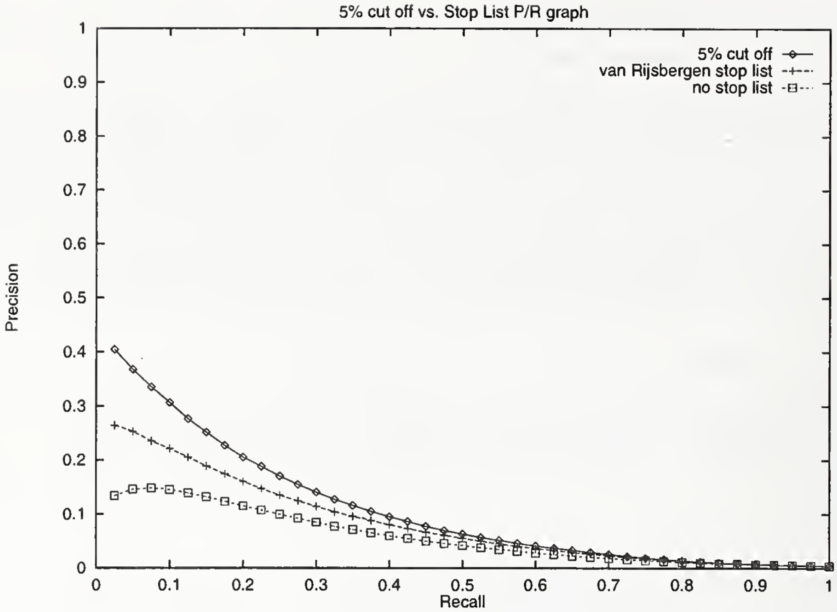


Figure 2: Precision and recall figures with different stop lists.

more than 70% of documents, and so on down to 2.5% of documents. For each stop list, the modified TREC-B collection was re-indexed, a retrieval run was performed and recall precision figures were obtained for each run. In addition two extra runs where performed where no stop list was used and a standard stop list from Van Rijsbergen [vR79] was used. The graph in figure 2 shows a selection of these runs.

As can be seem from the graph, precision is improved at all recall levels when a standard stop list is used instead of no stop list. If a stop list containing terms occurring in more than 5% of the documents is used however, there is a further uniform improvement in precision. The terms in this stop list account for around 50% of the term occurrences in the collection, as can be seen from Table 2. Although not plotted in this graph, it was found that using a larger stop list reduced precision.

From this experiment, it was concluded that the 5% stop list should be used when indexing the TREC-B collection so as to improve the speed of the RbLI process.

Initial number of occurrences	12,594,371
Occurrences after 5% stop list	6,902,676
Number of words in 5% stop list	253

Table 2: Effects of a 5% stop list

6 Evaluating Retrieval by Logical Imaging using the TREC-B document collection

Using the results reported in the previous section, we tried to perform a few experiments using the improved RbLI software on the *TREC-B document collection* for the *ad hoc* track. However, we soon run into a lot of small technical problems, the kinds of problems that almost all first time TREC participants experienced. Our lack of experience in dealing with large document collections together with the complexity of the RbLI model, made it impossible to have retrieval results ready for the TREC deadline.

In August 1995, when it became clear that the RbLI experiments would not be ready in time for the TREC-4 deadline, we decided to use a more classical IR system we already had developed in Glasgow to produce the retrieval results for the "ad hoc" track to send to TREC-4. We thought that we could later use the results of this system as a benchmark for the RbLI results, when these would be ready. The *glair* result set was then quickly generated using the system described in the following section and submitted instead of the RbLI results, that were not yet available.

6.1 The *glair* benchmark

The benchmark system we adopted for comparison with RbLI is "text book" IR system. It is based on the classical $tf \cdot idf$ retrieval strategy. Terms found in documents and queries have first their case normalised, then, any of these terms appearing in a stop list (taken from Van Rijsbergen [vR79] [2]) were removed. The remaining terms were suffix stripped using the Porter

[2] The stop list experiments reported in the previous section had not been carried out at this stage.

stemmer [Por80]. Document terms were weighted using the $tf \cdot idf$ weighting scheme. The *idf* formula has already been defined in Section 3, tf is defined as in [FB92]:

$$tf_{i,j} = \frac{\log(freq_{i,j} + 1)}{\log(length_j)}$$

where $freq_{i,j}$ is the frequency of term t_i in document d_j, and $length_j$ is the number of unique terms in document d_j.

The $tf \cdot idf$ retrieval strategy simply evaluates the product of the two components and ranks the documents in the collection based on a score. The score for each document is calculated by summing the $tf \cdot idf$ weights of any query terms found in that document.

We submitted to the TREC-4 Conference (TREC-B) the results of this system with low expectations. However, the results we achieved were not bad at all, as is summarised in the following excerpt from the official TREC-4 results:

```
Queryid (Num):       all  glair1
Total number of documents over all queries
    Retrieved:    49000
    Relevant:      2480
    Rel_ret:       1703
Interpolated Recall - Precision Averages:
    at 0.00       0.6321
    at 0.10       0.4332
    at 0.20       0.3085
    at 0.30       0.2507
    at 0.40       0.1747
    at 0.50       0.1453
    at 0.60       0.1035
    at 0.70       0.0744
    at 0.80       0.0543
    at 0.90       0.0393
    at 1.00       0.0229
Average precision (non-interpolated) over all rel docs
                  0.1819
Precision:
  At    5 docs:   0.3633
  At   10 docs:   0.3122
  At   15 docs:   0.2857
  At   20 docs:   0.2612
  At   30 docs:   0.2313
  At  100 docs:   0.1422
  At  200 docs:   0.0965
  At  500 docs:   0.0566
  At 1000 docs:   0.0348
R-Precision (precision after R (= num_rel for a query)
docs retrieved):
    Exact:        0.2170
```

The system, in the context of the TREC-B only participants, gave the best performances in 10 queries, and the worst performances in 6 queries out of the 49 used in TREC-B. Its overall performance was well above the median value of the average precision.

6.2 The RbLI results

Unfortunately, at the time of writing, we were still not able to obtain performance figures for RbLI. A few unexpected problems and some "bugs" in the RbLI software make impossible for us to have retrieval results from RbLI that can be compared with those obtained by our benchmark. Despite that, we think that this paper could provide a proof of our efforts toward using a complex logical model of IR on a large collection of documents, a task very rarely attempted by other researchers working on logical models for IR. We hope we set an example.

We will present the results of the use of RbLI on the TREC-B collection as soon as they are obtained.

7 Conclusions

We have been told by others that there is a tradition that "TREC first timers" fail to get their planned experiments done by the required deadline. We unfortunately have done nothing to change this.

Trying to implement RbLI on the TREC collection is proving to be a compromise between the theoretical purity demanded by the model, and the implementation problems posed by a collection of the size of TREC-B. We have found this compromise to a be a driving force in revealing other areas of work to be investigated. Therefore, experimental results aside, we regard our first participation in TREC as having been beneficial.

Acknowledgments

Most part of the work reported in this paper was supported by the ESPRIT Project *FERMI* (BRA 8134) on the "Formalisation and Experimentation of the Retrieval of Multimedia Information".

References

[Bru93] P.D. Bruza. *Stratified Information Disclosure: a synthesis between Hypermedia and Information Retrieval.* Phd thesis, Katholieke Universiteit Nijmegen, The netherlands, 1993.

[CC92] Y. Chiaramella and J.P. Chevallet. About retrieval models and logic. *The Computer Journal*, 35(3):233–242, 1992.

[Cro94] C.B. Cross. Eliminative bayesianism and probability revision. Unpublished paper, August 1994.

[Cv95] F. Crestani and C.J. van Rijsbergen. Probability kinematics in information retrieval. In *Proceedings of ACM SIGIR*, pages 291–299, Seattle, WA, USA, July 1995.

[CvR95] F. Crestani and C.J. van Rijsbergen. Information Retrieval by Logical Imaging. *Journal of Documentation*, 51(1):1–15, 1995.

[FB92] W.B. Frakes and R. Baeza-Yates, editors. *Information Retrieval: data structures and algorithms.* Prentice Hall, Englewood Cliffs, New Jersey, USA, 1992.

[G̈88] P. Gärdenfors. *Knowledge in flux: modelling the dynamics of epistemic states.* The MIT Press, Cambridge, Massachusetts, USA, 1988.

[HC68] G.E. Hughes and M.K. Cresswell. *An Introduction to Modal Logic.* Muthuen and Co. Ltd, London, UK, 1968.

[Hil65] D.J. Hillman. Topology and document retrieval operations. Studies of theories and models of information storage and retrieval, Lehigh University, July 1965.

[Kri71] S.A. Kripke. Semantical considerations on modal logic. In L. Linsky, editor, *Reference and modality*, chapter 5, pages 63–73. Oxford University Press, Oxford, UK, 1971.

[Lew81] D. Lewis. Probability of conditionals and conditionals probabilities. In W.L. Harper, R. Stalnaker, and G. Pearce, editors, *Ifs*, The University of Western Ontario Series in Philosophy of Science, pages 129–147. D.Reidel Publishing Company, Dordrecht, Holland, 1981.

[Nie88] J.Y. Nie. An outline of a general model for information retrieval. In *Proceedings of ACM SIGIR*, pages 495–506, Grenoble, France, June 1988.

[Nie89] J.Y. Nie. An Information Retrieval model based on Modal Logic. *Information Processing & Management*, 25(5):477–491, 1989.

[Por80] M.F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[RS76] S.E. Robertson and K. Sparck Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27:129–146, May 1976.

[Sal68] G. Salton. *Automatic information organization and retrieval.* Mc Graw Hill, New York, 1968.

[Sta81] R. Stalnaker. Probability and conditionals. In W.L. Harper, R. Stalnaker, and G. Pearce, editors, *Ifs*, The University of Western Ontario Series in Philosophy of Science, pages 107–128. D.Riedel Publishing Company, Dordrecht, Holland, 1981.

[vR77] C.J. van Rijsbergen. A theoretical basis for the use of cooccurence data in Information Retrieval. *Journal of Documentation*, 33(2):106–119, June 1977.

[vR79] C.J. van Rijsbergen. *Information Retrieval.* Butterworths, London, second edition, 1979.

[vR86] C.J. van Rijsbergen. A non-classical logic for Information Retrieval. *The Computer Journal*, 29(6):481–485, 1986.

[vR89] C.J. van Rijsbergen. Toward a new information logic. In *Proceedings of ACM SIGIR*, Cambridge, USA, June 1989.

Automatic Word Similarity Detection for TREC 4 Query Expansion

Susan Gauch and Meng Kam Chong
sgauch@eecs.ukans.edu
Electrical Engineering and Computer Science
University of Kansas

ABSTRACT

Accessing online information remains an inexact science. While valuable information can be found, typically many irrelevant documents are also retrieved and many relevant ones are missed. Terminology mismatches between the user's query and document contents is a main cause of retrieval failures. Expanding a user's query with related words can improve search performance, but the problem of identifying related words remains.

This research uses corpus linguistics techniques to automatically discover word similarities directly from the contents of the untagged TREC database and to incorporates that information in the PRISE information retrieval system. The similarities are calculated based on the contexts in which a set of target words appear. Using these similarities, user queries are automatically expanded, resulting in conceptual retrieval rather than requiring exact word matches between queries and documents.

1. INTRODUCTION

Expanding a user's query with related terms can improve search performance. Relevance feedback systems, where related terms come from the contents of user-identified relevant documents, have been shown to be quite effective (Harman 1992). Our earlier work showed that an expert system which automatically reformulated Boolean queries by including terms from an online thesaurus was able to improve search results (Gauch and Smith 1991; Gauch and Smith 1993) without requiring relevance judgments from the user. Some systems (Anick, Brennan et al. 1990) present related terms to the user and allow them to selectively augment the query. However, the latter two approaches require the presence of an online thesaurus whose words closely match the contents of the database.

Where can such a thesaurus come from? In some cases, it is hand-built (Gauch and Smith 1991), a time-consuming and ad hoc process. In other cases, the thesaurus is an online version of a published thesaurus or semantically coded dictionary (Liddy and Myaeng 1993). However, an online published thesaurus or dictionary will have serious coverage gaps if used for technical domains which have their own distinct sublanguages. Because of ambiguity, this type of thesaurus may also be difficult to use with a database of general English documents because they show all possible classifications for a word when only one or a few senses may be actually present in the database.

Our system to automatically discovers related words directly from the contents of a textual database and incorporates that information in a traditional information retrieval system. We modified and applied one particular techniques from the field of corpus linguistics which seemed particularly well-suited for this task. HNC's MatchPlus system (Gallant, Hecht-Nielsen et al. 1993) has a similar approach, however, they use neural networks to identify features which are used to index documents rather than using the words themselves. In contrast, we index documents by their words and identify related words which can be used for query expansion. With our approach, it is possible to provide query expansion on top of pre-indexed collections.

2. SYSTEM ARCHITECTURE

Our goal is to incorporate the results of the corpus analysis into an existing retrieval engine. For this purpose, we evaluated three freely available text retrieval engines: freeWAIS, SMART and PRISE. These three retrieval engines are all vector space model which use inverted file database structures. Each retrieval engine was evaluated on three document collections, a database of biology paper abstracts, the standard CACM rest collection and the TREC3 database. Based on the results, the PRISE system was selected for our TREC4 entry. It was modified to allow it to expand queries based on the similarity matrices, search the database with the expanded queries, and return the top 1000 documents for each query.

We participated in category B, which is evaluated based on two collections: the Wall Street Journal (WSJ) and the San Jose Mercury (SJM). Combined, the databases are 0.5 GB in size and contain 164777 documents. Indexing the database took approximately 11 hours on a shared Sun SPARC 10. Both databases are in SGML format, which is the input format for PRISE. The stemming function of the PRISE system is turned off, since the automatic query expansion phase will introduce words which share a stem, if their usage in the database is similar enough.

3. CORPUS LINGUISTICS TECHNIQUE

Methods that work with entirely untagged corpora have recently been developed which show great promise (Brill and Marcus 1992; Finch and Chater 1992; Hearst, 1992, Myaeng and Li 1992; Schütze 1992). Using a much more fine-grained approach than traditional automatic thesaurus construction techniques, word-word similarities are automatically calculated based on the premise that words which occur in similar contexts are similar. These techniques are particularly useful for specialized text with specialized vocabularies and word-use, for which there are no adequate online dictionaries. They are also appropriate for general English corpora since a general online dictionary may show many senses for a common word where only one or a few actually are used in a given corpus.

We have modified a corpus linguistics approach (Finch and Chater 1992) that takes into account both the relative positions of the nearby context words as well as the mutual information (Church and Hanks 1990) associated with the occurrence of a particular context word. We have applied this to a sample of the TREC4 database to calculate *a priori* the similarities of a subset of the words in the database, called the target words.

3.1 Similarity Calculation

Similar to (Finch and Chater 1992), the context vector for a word (the *target word*) is a concatenation of the vectors describing the words observed in the preceding two positions and the following two positions (the *context positions*). Each position is represented by a vector corresponding to the occurrence of the 150 highest frequency words in the corpus (the *context words*), giving a 600-dimensional vector describing the context. Initially, the counts from all instances of a word form w_i are summed so that the entry in the corresponding context word position in the vector is the sum of the occurrences of that context word in that position for the corresponding target word form; it is the joint frequency of the context word.

Consider an example in which there are only five context words, {"a", "black", "dog", "the, "very"} and two sentences containing the target word "dog:

(1) The black dog barked very loudly.

(2) A brown dog barked very loudly.

Sentence	Context Position	Observed Word	Context Vector
1	-2	"The"	(0, 0, 0, 1, 0) 4th context word
	-1	"black"	(0, 1, 0, 0, 0) 2nd context word
	+1	"barked"	(0, 0, 0, 0, 0) not a context word
	+2	"very"	(0, 0, 0, 0, 1) 5th context word

Table 1. The context vectors for each of the 4 context positions around the occurrence of the target word "dog" in sentence 1.

The context vector for "dog" in sentence 1 is formed by concatenating the context vectors for each of the 4 context positions:

(0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1)

Similarly, the context vector for "dog" in sentence 2 would be:

(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1)

and the combined vector for the word "dog" would be:

(1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2)

Using 150 context words, 600-dimensional context vectors are created. Subsequently, 600-dimensional vectors of mutual information values, *MI*, are computed from the frequencies as follows,

$$MI(cw) = \log_2\left(\frac{Nf_{cw}}{f_c f_w} + 1\right)$$

This expresses the mutual information value for the context word *c* appearing with the target word *w*. The mutual information is large whenever a context word appears at a much higher frequency, f_{cw}, in the neighborhood of a target word than would be predicted from the overall frequencies in the corpus, f_c and f_w. The formula adds 1 to the frequency ratio, so that a 0 (zero) occurrence corresponds to 0 mutual information. When the mutual information vectors are computed for a number of words, they can be compared to see which words have similar contexts. The comparison we chose is the inner product, or cosine measure, which can vary between -1.0 and +1.0 (Myaeng and Li 1992).

Finally, to make the identification of the most highly similar terms to a given term more efficient, an auxiliary file is produced *a priori* from the similarity matrix. It stores, for each target word, the words and similarity values for all words with similarity above a given threshold. This is called the similarity lists.

3.2 Preprocessing the Database

For use by the corpus analysis program, the TREC4 database is sampled randomly and uniformly to get a representative sample (roughly 15%) of the whole database. The resulting WSJ sample was 29 MB. Words in the sample range in frequency from 1 to 280,004. There are 102,912 distinct words and 79619 words have frequency less than 6. The corresponding SJM sample is 33 MB and contains words with frequency from 1 to 334,507. There are 113,240 distinct words and 75,602 words have frequency less than 6. During preprocessing, the required fields of a document are extracted and all capitalized tokens are put in lower case. Then, a sentence tagging algorithm has been implemented to identify sentence breaks. Furthermore, the sentences are tokenized by separating the comma, full stop, semicolons etc. from the words.

We then create a file containing the list of words in each sample, sorted by frequency. From this list, the corpus program automatically selects a set of target words and a set of context words. These are used as input to the similarity calculation phase (details in section 3.2). The target words are those which will have their similarities calculated. For efficiency reasons we want to limit the number of words studied to around 10,000. It takes about 10 hours per sample to calculate the 10,000 x 10,000 similarity matrix. However, only words in the target words will be able to be expanded should they appear in a query. Therefore, it is important to select a set of target words that will best match the content bearing words which appear in queries. To help us select our target words well, we analyzed the distribution of the words that used in the 150 TREC3 queries.

Based on our analysis, we found that 50% of the query words are the words which have frequency count greater than 2000 in both databases. These words are like "the", "an", "often" and etc. Those are the words that do not contain a lot of information, but do give a lot of context information. So, these words have been selected as the context words. Those words which fall between frequency count 100 and 2000 are the words that appear to be information-bearing yet are frequent enough to be studied statistically. Therefore, these are the words that have been selected as the target words. Thus, we select the target words as those whose frequencies are between 0.03% and 0.8% of the most frequent word in the sample. Context words are the words which have frequency count greater than 0.8% most frequent word.

3.3 Corpus Analysis Modification for Large Databases

In the original program, all information was stored in memory. We modified the program so that information is stored into randomly accessible binary files. This modification does improve the usage of memory and the space complexity becomes n versus n^2 in the original version. However, some of the speed of the program has been sacrificed. Another modification was to create an auxiliary file which stores, for each target word, a sorted list of all words whose similarity is greater than a given threshold. These *similarity lists* are also written in binary format, which allows reduces the memory usage of the modified retrieval engine.

4. TREC4 EXPERIMENT

4.1 Matrix Selection Algorithm

Since there are two databases (WSJ and SJM) in this project, two similarity matrices are generated, one for each database. We chose to do this because each of the databases has a different domain of interests, resulting in different word usage. However, this means that it is very important to select the correct matrix to expand a given query. Thus, for each query, we first calculate which database it best matches using a simple calculation based on the sum of the frequencies of the query words in each database. Whichever database maximizes the sum has its corresponding similarity matrix selected to expand the query.

This technique was tested with queries 101-200 from TREC3. The results are promising. Queries 101-150 are the queries that used to query both databases. The simple calculation assigned 28 of these 50 queries are to the SJM database. In contrast, queries 151-200 are the queries intended for the WSJ database. By applying the same technique to these 50 queries, 41 queries out of the 50 queries are identified as be related to database WSJ.

4.2 PRISE Retrieval Engine Modification

PRISE has been modified at the point just before a query is passed to the search engine. At that point, the query is expanded with the appropriate similarity matrix. The modified PRISE uses five files besides in addition to the regular inverted files. These are: the two similarity lists, frequency counts for the two databases, and a "threshold" file. The threshold file specifies the minimum similarity score necessary for a word to be included by query expansion. In addition, the display routine has been modified to display the similarity scores of a document - query matches

4.3 Query Expansion and Comparison of Matrices

Once the three systems were evaluated and the PRISE system chosen, we ran a series of experiments with the TREC3 queries to tune the corpus analysis program. There are four main parameters that can be adjusted: the list of target words, the list of context words, and the window size (the window around target words that is used to characterize the contexts in which they appear), and the similarity threshold that is used during query expansion. Due to time limitations, we were unable to run as many experiments as we would have liked, only evaluating the effects of changing the window size and similarity threshold used during query expansion. There were 4817 and 5205 target words for the WSJ and SJM, respectively. One hundred ninety-eight context words were used for the SJM database, and 261 context words for the WSJ database. The following table shows the 11 point average for the experiments:

Window Size	Threshold	11 point average
5	0.30	0.0648
5	0.45	0.0820
5	0.55	0.1064
5	0.57	0.1059
7	0.30	0.0768
7	0.38	0.0802
7	0.40	0.1002
7	0.43	0.1070
7	0.44	0.1068
7	0.45	0.1056
7	0.47	0.1059
Without Expansion		0.1037

Table 2. The 11 point averages based on different context window sizes and query expansion similarity thresholds.

Based on these results, for TREC 4, similarity matrices were constructed using a window of 7 and 0.43 was used as the threshold during query expansion.

4.4 TREC4 Results

The TREC4 queries were shorter than those in the previous TRECs. Thus, I expect that query expansion will be even more important. Here is a table summarizing our results compared to the entries in the ad hoc category:

Topic	# Rel.	Best	Median	Worst	KU1
202	178	66	62	52	66
203	20	8	5	1	5
204	140	53	35	5	24
205	63	5	4	2	5
206	16	2	2	0	0
207	43	33	30	27	33
208	26	10	9	2	8
209	27	13	7	2	2
210	5	2	2	0	0
211	124	21	19	15	21
212	78	39	36	28	32
213	8	1	1	1	1
214	1	1	0	0	1
215	83	41	34	20	32
216	23	11	9	8	8
217	14	6	3	2	3
218	40	16	11	8	16
219	64	22	18	11	17
220	3	3	2	2	3
221	83	35	34	29	31
222	16	9	8	7	7
223	97	52	50	43	52
224	79	25	19	12	19
225	38	14	13	10	13
226	66	8	2	0	8
227	115	33	27	12	25
228	32	8	7	4	7
229	7	5	3	0	2
230	68	22	14	2	10
231	4	4	4	3	4
232	4	0	0	0	0
233	64	4	3	2	3
234	6	4	3	3	4
235	59	18	12	3	12
236	0	0	0	0	0
237	121	38	33	21	33
238	70	11	5	1	11
239	32	7	5	4	4
240	89	24	6	3	3
241	33	2	2	1	1
242	18	6	3	0	2
243	32	12	10	5	10
244	170	62	49	26	26
245	29	9	6	3	5
246	118	29	25	8	23
247	14	7	7	3	3
248	20	5	5	0	5
249	10	2	0	0	2
250	30	12	11	8	12

4.5 Discussion

The results need to be analyzed to compare the results with expansion to those produced by the unmodified PRISE system. I suspect that the quality of our results at this time are primarily due to the quality of the PRISE system. The main problem is that our target words (those which can be expanded) do not include the important words in the query. For example, on topic 203, we retrieved only 5 relevant documents out of a possible 20, which was the median, whereas the best system retrieved 8. This was a query which was not much expanded by our system:

Original: *What is the economic impact of recycling tires?*

After expansion: *what 1.000000 is 1.000000 the 1.000000 economic 0.316406 political 0.156178 financial 0.154311 nuclear 0.126436 civil 0.124980 foreign 0.121690 impact 1.000000 of 1.000000 recycling 1.000000 tires 1.000000*

The only word expanded is economic, which was expanded with *political, financial, nuclear, civil,* and *foreign* with decreasing similarity weight. While these are reasonably similar words, the key words in the query, *recycling* and *tires*, were not target words and thus were not expanded at all.

Our next tasks are to do a better job of identifying the appropriate set of target words and further tuning the corpus analysis program to improve our results.

BIBLIOGRAPHY

Anick, P.G., Brennan, J.D., Flynn, R.A., Hanssen, D.R., Alvey, B., and Robbins, J.M. (1990). A Direct Manipulation Interface for Boolean Information Retrieval via Natural Language Query. Proc. 13th Ann. International ACM SIGIR Conf., (pp. 135-150). Brussels, Belgium: ACM Press.

Brill, E., & Marcus, M. (1992). Tagging an Unfamiliar Text with Minimal Human Supervision. In AAAI Fall Symposium Series: Probabilistic Approaches to Natural Language (Working Notes), (pp. 10-16). Cambridge, MA.

Church, K. W., & Hanks, P. (1990). Word Association Norms, Mutual Information and Lexicography. Computational Linguistics, 16(1), 22-29.

Finch, S., & Chater, N. (1992). Bootstrapping Syntactic Categories Using Statistical Methods. In W. Daelemans & D. Powers (Ed.), Proc. 1st SHOE Workshop, (pp. 229-235). Tilburg U., The Netherlands.

Gallant, S.I., Hecht-Nielsen, R., Caid, W., Qing, K., Carleton, J., and Sudbeck, C.D. (1993). HNC's MatchPlus System, 1st Text Retrieval Conf. (TREC-1), (pp. 107-111). NIST #500-207.

Gauch, S., & Smith, J.B. (1993). An Expert System for Automatic Query Reformulation. J. of the Amer. Society of Inf. Sci., 44 (3), 124-136.

Gauch, S., & Smith, J.B. (1991). Search Improvement via Automatic Query Reformulation. ACM Trans. on Information Systems, 9 (3), 249-280.

Harman, D. (1992). Relevance Feedback Revisited. Proc. 15th Ann. International ACM SIGIR Conf., (pp. 1-10). Copenhagen, Denmark: ACM Press.

Hearst, M.A. (1992) "Automatic Acquisition of Hyponyms from Large Text Corpora," Proc. 14th Intern'l Conf. on Computational Linguistics, Nantes, France, July.

Liddy, E.D., & Myaeng, S.H. (1993). DR-LINK's Linguistic-Conceptual Approach to Document Detection, 1st Text Retrieval Conf. (TREC-1), (pp. 113-129). NIST #500-207.

Myaeng, S. H., & Li, M. (1992). Building Term Clusters by Acquiring Lexical Semantics from a Corpus. In Y. Yesha (Ed.), CIKM-92, (pp. 130-137). Baltimore, MD: ISMM.

Schütze, H. (1992). Context Space. In AAAI Fall Symposium Series: Probabilistic Approaches to Natural Language (Working Notes), (pp. 113-120). Cambridge, MA:

Report on the TREC-4 Experiment: Combining Probabilistic and Vector-Space Schemes

Jacques Savoy, Melchior Ndarugendamwo, Dana Vrajitoru

Faculté de droit et des sciences économiques
Université de Neuchâtel
Pierre-à-Mazel 7
CH - 2000 Neuchâtel (Switzerland)

Summary

This paper describes and evaluates a retrieval scheme combining the OKAPI probabilistic retrieval model with various vector-space schemes. In this study, each retrieval strategy represents both queries and documents using the same set of single terms; however they weight them differently. To combine these search schemes, we do not apply a given combination operator on the retrieval status nor the rank of each retrieved record (e.g., sum, average, max., etc.). We think that each retrieval strategy may perform well for a set of queries and poorly for other requests. Thus, based on a given query's statistical characteristics, our search model first selects the more appropriate retrieval scheme and then retrieves information based on the selected search mechanism. Since the selection procedure is done before any search operation, our approach has the advantage of limiting the search time to one retrieval algorithm instead of retrieving items using various retrieval schemes, and then combining the given results.

In particular, this study addresses the following questions: (1) can the statistical characteristics of a query be good predictors in an automatic selection procedure; (2) faced with the relatively high retrieval effectiveness achieved by the OKAPI model, can various vector-space schemes further improve the retrieval performance of the OKAPI approach, and (3) can the learning results obtained with one tested collection (WSJ) be valid for another corpus (SJMN)?

Participation: Category: B Query: ad-hoc, fully automatic

Introduction

There are many reasons for using multiple sources of evidence [Katzer et al. 1982] [Tenopir 1985]. Firstly, the studies comparing the same document representation scheme do not always produce the same result, because they are not based on the same domain of knowledge, they use different collections of documents, different stemming algorithms, etc. Secondly, when we compare the performance (e.g., recall and precision) of different representations, none is found to perform well for all criteria. Thirdly, a comparison of mean performances among various search strategies reveals that when a difference occurs, it is small. Fourthly, even for representations considered similar, such as "abstract" and "title and abstract", the overlap between pairs of representations is very low (around 35%); therefore, one cannot assert that distinct document representations can be considered as equivalent (see also [Noreault et al. 1981] about representations built on controlled vs. free-text vocabularies). Fifthly, when studying the overlap in retrieved items by various searchers searching the same question, Saracevic & Kantor [1988, p. 204-207] find that the intersection is relatively low (e.g., for around 78.7% of all items retrieved, the degree of agreement is less than 25%). This fact cannot be explained by a low search term overlap however:

> "the conclusion that different searchers for the same question see and interpret different things in a question, represent them by different linguistic and/or logical constructs, and retrieve different things from a file." [Saracevic & Kantor, 1988, p. 204]

Finally, the analysis of various TREC experiments [Harman 1994] demonstrates that a given retrieval scheme may perform very well for some queries and poorly for other ones. Therefore, overall statistics, like the average precision, may hide performance irregularities among requests when comparing different retrieval schemes. To overcome these problems, the combination of retrieval schemes seems to be a necessity.

The integration of multiple sources of information (especially provided by other retrieval information schemes) is currently analyzed in two different contexts. These retrieval strategies may operate on the same collection (data fusion problem), on the one hand, and, on the other hand, they may retrieve items for a given request from different corpora or various information servers (collection fusion problem). This latter question involves the merging the retrieval results of searches on independent collections into an effective, single ranked list [Voorhees et al. 1995, April], [Voorhees et al. 1995, July], and particularly, this problem appears in distributed systems, such as the WWW.

In this study, we are concerned with the data fusion problem, where an important question does arise : is it pertinent to consider multiple retrieval schemes operating on the same collection? The answer seems to be positive. For example, Saracevic & Kantor [1988, p. 204-207] demonstrate that the odds of a document being relevant to a given request increases monotonically with the number of search strategies that retrieve this record. Moreover, for items retrieved only once, the relevance odds decrease about 8 to 10. Other studies reveal that one can increase the performance of a retrieval system by using multiple document surrogates, various query formulations or by combining multiple search schemes (e.g., [Fox et al. 1988], [Turtle & Croft 1991], [Thompson 1993], [Fox et al. 1993], [Belkin et al. 1993], [Belkin et al. 1994], [Bartell et al. 1994], [Fox & Shaw 1994], [Shaw & Fox 1995], [Lee 1995]).

Traditionally, in studying the data fusion problem, the retrieval engine first find the retrieved set of each retrieval scheme and then defines an appropriate merging function. For this combination, we may consider the rank of the retrieved records and / or their retrieval status value. In this vein, Fox et al. [1993], [Fox & Shaw 1994], [Shaw & Fox 1995] show various ad hoc schemes in combining the p-norm model and vector-processing strategies. Of course, these retrieval schemes may be based on different indexing strategies of the same collection. In this latter case, the retrieval status values may not have a range of possible similar values, leading to a more complex combination problem (see also [Lee 1995]). Moreover, we may need to weight each retrieval scheme (or define a predicting relevance based on the relative merit of each single search strategy) based on previous relevance assessments.

In this study, we do not consider really different requests (e.g., Boolean and natural language queries) or document representations (e.g., single terms vs. phrase indexing strategies). Each query or document is represented by the same set of single terms and the weight assigned to each keyword may vary according to each retrieval scheme. Since our selection procedure takes place before any retrieval operation, our retrieval cost is limited to only one search algorithm instead of summing (at least with von Neuman architecture), the computation time required by each single search scheme [Fox et al. 1993], [Fox & Shaw 1994], [Shaw & Fox 1995], [Lee 1995].

Our retrieval procedure can be viewed as a trial and error process [Swanson 1977]. Moreover, in this spirit, if our automatic selection procedure fails for a given request to choose the best retrieval scheme, the user may have the opportunity to select a more appropriate search strategy (at least, in the user's opinion).

The rest of this paper is organized as follows. The next section presents an overview of both the vector-space and the OKAPI probabilistic models. The second section describes the basic principles of the k-Nearest-Neighbors method (k-NN) used to select the more appropriate retrieval scheme based on statistical features of each query.

1. Retrieval Models

To define a retrieval model, we must explain how documents and queries are represented and how these representations are compared to produce a ranked list of retrieved items. In this experiment, the indexing procedure done by the SMART system is fully automatic and based on a single term only.

To achieve this goal, each topic was indexed according to the content of its Descriptive (<desc>) section only. For each document, Text (<text>) section as well as Subtitle (<ST>), Headline (<HL>), and Summary (<LP>) sections were indexed for the WSJ collection (<leadpara>, <text> and <headline> for the SJMN corpus). All other subsections were removed, and, in particular, the title, the narrative and the concept section of each topic (see Table 1).

Collection	Section
WSJ	<desc>, <text>, <st>, <hl>, <lp>
SJMN	<leadpara>, <text>, <headline>
Query	<desc>

Table 1: Selected Sections Used to Represent Documents and Queries

	Indexing Weight
NNN	$w_{ij} = tf_{ij}$
ANN	$w_{ij} = 0.5 + 0.5 \cdot \frac{tf_{ij}}{\max tf_{ik}}$
LNC	$w_{ij} = \frac{\log(tf_{ij})+1}{\sqrt{\sum_{k=1}^{t} (\log(tf_{ik})+1)^2}}$
LTC	$w_{ij} = \frac{[\log(tf_{ij})+1] \cdot idf_j}{\sqrt{\sum_{k=1}^{t} ([\log(tf_{ik})+1] \cdot idf_k)^2}}$

Table 2: Indexing Weighting Schemes

As shown in Section 1.1, various weighting schemes can be used within the vector-space model, leading to different retrieval effectiveness. Moreover, this section demonstrates that the length of the query may play an important role in the retrieval performance. Section 1.2 describes the probabilistic model OKAPI based on a different weighting and matching strategy.

1.1. Evaluation of the Vector Space Model

To represent each document and each query by a vector of weighted keywords, the vector-space model suggests various weighting schemes. To select the more appropriate one, we have conducted a set of experiments based on different weighting formulas. To assign an indexing weight w_{ij} reflecting the importance of each single-term T_j, $j = 1, 2, ..., t$, in a document D_i, we may use one of the equations shown in Table 2. In this table, tf_{ij} depicts the frequency of the term T_j in the document D_i (or in the request), n represents the number of documents D_i in the collection, df_j the number of documents in which T_j occurs, and idf_j the inverse document frequency (log $[n/df_j]$).

To normalize each indexing weight between 0 and 1, we may consider the cosine normalization (see LNC formula in Table 2), or we may also take account of the distribution of each indexing term in the collection by giving a higher weight to sparse words and lower importance to more frequent terms (idf component in LTC formula in Table 2).

To define the retrieval status value (RSV) of each document D_i, the vector-space model uses the following equation:

$$RSV_{VSM}(D_i) = \sum_{k=1}^{q} w_{ik} \cdot w_{qk} \tag{1}$$

where w_{ik} represents the indexing term weight of T_k in a document D_i, w_{qk} the keyword search weight of T_k in the current query and q the number of search keyword in the request.

In Table 3, we compare the retrieval effectiveness achieved using various indexing schemes and three different query formulations. In this table and in the following tables, precision is measured at eleven standard recall values (from 0.0 to 1.0) for all queries (#1 through #200), and then averaged to form our retrieval effectiveness measure. The numbers in parenthesis indicate the percent of change computed by the system based on the baseline solution. To decide whether a search strategy is better than another, a difference of at least 5% in average precision is generally considered significant and, a 10% difference is considered very significant.

For a long query, the Descriptive, Narrative and Title sections of the topic description were used to build the request vector, while the shortest query form is built only with the Descriptive section (the new request form for TREC'4). The middle column shows the retrieval performance achieved by using both the Descriptive and the Narrative sections.

	Average Precision (% change)		
Model \ Query Form	<desc> (baseline)	<desc> and <narr>	<desc>, <narr> and <title>
Vector-Space Model			
doc = NNN, query = NNN	4.59	7.03 (+53.2)	9.42 (+105.2)
doc = ANN, query = ANN	9.41	10.22 (+8.6)	13.07 (+38.9)
doc = LNC, query = LNC	9.64	19.79 (+105.3)	22.48 (+133.2)
doc = LTC, query = LTC	15.69	23.26 (+48.2)	25.63 (+63.35)
doc = LTC, query = LNC	12.10	21.11 (+74.5)	23.58 (+94.9)
doc = LNC, query = LTC	17.04	27.42 (+60.8)	29.95 (+75.8)
Probabilistic Retrieval Model			
OKAPI	22.56	33.01 (+46.3)	32.49 (+44.0)

Table 3: Evaluation of Individual Retrieval Schemes (WSJ Collection)

From the data shown in Table 3, we may find that:

1. for all request representations, the OKAPI probabilistic model achieves the highest retrieval performance;
2. when considering only the vector-space model, the best result is obtained when using the LNC-LTC strategy.
3. when the request representation includes more information about the user's information need, the retrieval performance is enhanced. The only exception to this rule is the OKAPI model which achieved similar performance using a long or a medium size query formulation (33.01 vs. 32.49 (-1.6%)).

1.2. OKAPI Probabilistic Model

Based on TREC'3 results [Robertson et al. 1995b], the OKAPI probabilistic approach presents a very attractive retrieval model. This model is based on the combination of two probabilistic schemes using information about term frequency both in the request and in the document, and incorporating a correction factor to account for document length.

The OKAPI probabilistic model is based on: (1) the weighting of the search term as a traditional probabilistic model (represented by the component $w^{(1)}$); (2) the frequency of the indexing term (component tf_{ik}); (3) the frequency of the search term (component tf_{qk}), and (4) a length correction factor (component avdl) taking account of document surrogate length. Schematically, the computation of the retrieval status value of each document is expressed as:

$$RSV_{BM}(D_i) = \text{CorrectionFactor} + \sum_{k=1}^{q} w_{ik} \cdot w_{qk}$$

A formal derivation of w_{qk} assigned to each search keyword is obtained in the probabilistic retrieval model by making use of Bayes' theorem and term-independence assumption postulating that the index terms occur independently in the relevant and nonrelevant document (for details see [van Rijsbergen 1979, Chapter 6]). In this case, the weight w_{qk} is evaluated according to Formula 2.

$$w^{(1)} = \log\left[\frac{r_{qk}}{1 - r_{qk}}\right] + \log\left[\frac{1 - s_{qk}}{s_{qk}}\right] = \log\left[\frac{n - df_k}{df_k}\right] + c \quad \text{with } c = \log\left[\frac{r_{qk}}{1 - r_{qk}}\right] \qquad (2)$$

in which r_{qk} (s_{qk}) expresses the conditional probability of knowing the document is relevant (nonrelevant), its representative containing the index term T_k.

The combined approach, called BM25, is based on the following evaluation of the retrieval status value:

$$RSV_{BM25}(D_i) = k_2 \cdot l_q \cdot \frac{avdl - l_i}{avdl + l_i} + \sum_{k=1}^{q} s_1 \cdot \frac{tf_{ik}^c}{K^c + tf_{ik}^c} \cdot w^{(1)} \cdot s_3 \cdot \frac{tf_{qk}}{k_3 + tf_{qk}} \qquad (3)$$

$$\text{with } K = k_1 \cdot \left[(1 - b) + b \cdot \frac{l_i}{avdl}\right]$$

where avdl means the average document surrogate length, l_i (l_q) represents the document representative length (query length, respectively), k_1, k_2, k_3, s_1, s_3 are unknown constants, and $w^{(1)}$ is estimated by Equation 2.

Parameter	$BM25_{WSI}$	$BM25_{CACM}$	$BM25_{CISI}$
$k_1 =$	2	2	2
$k_2 =$	0	0	0
$k_3 =$	∞	5	∞
b =	0.75	0.375	0.125
c =	1	1	1
$s_1 =$	2	2	2
$s_3 =$	1	5	1
$r_{qk} =$	0.5	0.6	0.5
avdl =	241.332	33.9919	67.6062
n =	173,252	3,204	1,460

Table 4: Parameter Setting for OKAPI Probabilistic Model

In order to define an "optimal" parameter setting for the BM25 model, we have to conduct a set of experiments based on the CACM and CISI test-collections [Savoy 1995]. The results are depicted in Table 4. However, in our current context, we have set our retrieval scheme according to the parameter values given by [Robertson et al. 1995b] (or $BM25_{WSJ}$ in Table 4).

2. Combination of Various Retrieval Models

In the previous section, we described two retrieval models that can be used to retrieve information from a textual database. However, it is known that a given search scheme may perform well for some requests but poorly for other ones. This phenomenon is clearly demonstrated by the analysis of variance described in [Tague-Sutcliffe & Blustein 1995] indicating that the variability attributable to the queries is much greater than those due to the different search strategies. However, in referring to the retrieval effectiveness shown in Table 3, we might ask whether we can really improve the particularly high performance achieved by the OKAPI probabilistic model.

To answer this question, we have designed a selection procedure which, based on query features, automatically selects the "best" single retrieval strategy for the current request. This problem can be viewed as a classification task within which a decision has to be made on the basis of currently available information. Various learning schemes can be conceived, such as Bayesian classifiers (linear or quadratic discriminant functions), k-Nearest-Neighbors (k-NN), Neural Networks, logistic regression, rule-based methods or Decision Trees [Weiss & Kulikowski 1991], [Michie et al. 1994].

For our purposes, we have chosen the k-NN method already used in IR studies in different contexts; for example, to define the number of documents to be selected from different information servers [Voorhees et al. 1995, April]. In this case, the similarity between queries is computed based on the keywords contained in the different requests and not based on their statistical characteristics. Moreover, Michie et al. [1994, p. 185] have demonstrated that the k-Nearest-Neighbors method performs similar to statistical methods which perform particularly well when the selected features are good predictors, and seem to perform better than rule-based or Decision Trees methods.

2.1. Is a Selection Procedure Really Useful?

From previous research, we can conclude that the combination of various retrieval schemes is a useful strategy for enhancing retrieval effectiveness. However, in our context, we do not really combine the retrieved items from different retrieval schemes, but our system tries to select a single retrieval based on the statistical characteristics of the current request. For a given request, this strategy has the advantage that only one single retrieval mechanism has to be computed.

However, our selection procedure must choose between one probabilistic retrieval strategy having a particularly high mean retrieval effectiveness and six vector-space schemes as shown in Table 5. Based on 200 queries, Table 5 depicts the average precision and, for each retrieval scheme, the number of best individual runs on a per query basis. Thus, for 132 queries out of 200, the best choice is the OKAPI model. It is interesting to note that the best vector-space approach (document = LNC, query = LTC) does not result in a significantly different number of best runs compared to other vector-space schemes.

Model	Average Precision	# best run	# best run
			δ = 2%
				# best run
				δ = 5%

Vector-Space Model				
1. doc = NNN, query = NNN	4.59	10	10	10
2. doc = ANN, query = ANN	9.41	13	13	12
3. doc = LNC, query = LNC	9.64	7	7	7
4. doc = LTC, query = LTC	15.69	24	22	21
5. doc = LTC, query = LNC	12.10	2	1	1
6. doc = LNC, query = LTC	17.04	12	12	12
Probabilistic Retrieval Model				
7. OKAPI	22.56	132	135	137
Combined Approach				
Using schemes 1, 2, 3, 4, 5, 6, 7		24.10	24.09	24.09

Table 5: Characteristics of Individual Retrieval Schemes (WSJ Collection)

From Table 5, one can see that:

1. the overall good performance of the OKAPI model hides some irregularities;
2. even a simple retrieval scheme like the vector-space model based on the NNN or ANN indexing scheme represents the best scheme for 13 requests out of 200 (or 11.5% of the cases);
3. the best vector-space scheme (document = LNC, query = LTC) does not represent the highest number of best runs for the vector-space model;
4. the optimal selection may enhance the OKAPI probabilistic retrieval model of around 7% (24.10 vs. 22.56).

However, one can argue that when a vector-space model reveals a better retrieval performance, the difference must be small compared to the OKAPI model. In response to this question, Table 5 depicts the number of best individual runs on a per query basis, provided that the difference δ is greater than 2% (or 5%) compared to the OKAPI model. The data shown in this table is clear: when a difference occurs, it is greater than 5% compared to the probabilistic retrieval strategy, or in other words, ignoring small differences it leads to a similar retrieval performance (24.09 vs. 24.10).

2.2. Principles of k-Nearest-Neighbors (k-NN)

The aim of our selection problem can be stated as follows (see Table 6):

- given a set of search scheme$_h$, h = 1, 2, ..., q;
- given a set of query features f_i, i = 1, 2, ..., p;
- given a set of query Q_j, j = 1, 2, ..., m for which we know both the values of each feature and the best retrieval scheme;
- find the optimum retrieval scheme$_j$ for a new request Q knowing the values of each of its features.

As described in the previous section, this study evaluates seven retrieval schemes. As query features, other studies have considered the number of terms in the request, the sum of the idf over all search keywords, the mean of these query search weights [Croft & Thompson 1984], [Croft & Thompson 1987]. In this study, we have retained the following seven statistical features:

1. the number of search included in the request;
2. the maximum term frequency (tf);
3. the tf mean;
4. the maximum inverse document frequency (idf);
5. the idf min;
6. the idf mean;
7. the idf median.

This list does not include the term frequency median (tf median) because this measure has a null standard deviation. Finally, based on this data, we must select a classification method to predict the retrieval scheme to apply to each new request. In this study, we have chosen a simple learning method, the k-NN method which works as follows.

For each new topic Q, the system must find the k nearest neighbors in the set of all existing request Q_j, j = 1, 2, ..., m. To define a suitable metric for this purpose, the Euclidean distance has been chosen. More precisely, the difference between the values of each feature is squared and summed over all features. The square root of this sum defines the actual Euclidean distance and the minimum distance indicates the nearest neighbor.

Query	Features Schemes				Search
Q1	x11	x12	...	x1p	scheme1
Q2	x21	x22	...	x2p	schemeq
...	...	...	...	...	...
Qj	xj1	xj2	...	xjp	scheme2
...	...	...	...	...	...
Qm	xm1	xm2	...	xmp	scheme1
New Q	x_1	x_2	...	x_p	?

Table 6: Example of a Classification Task

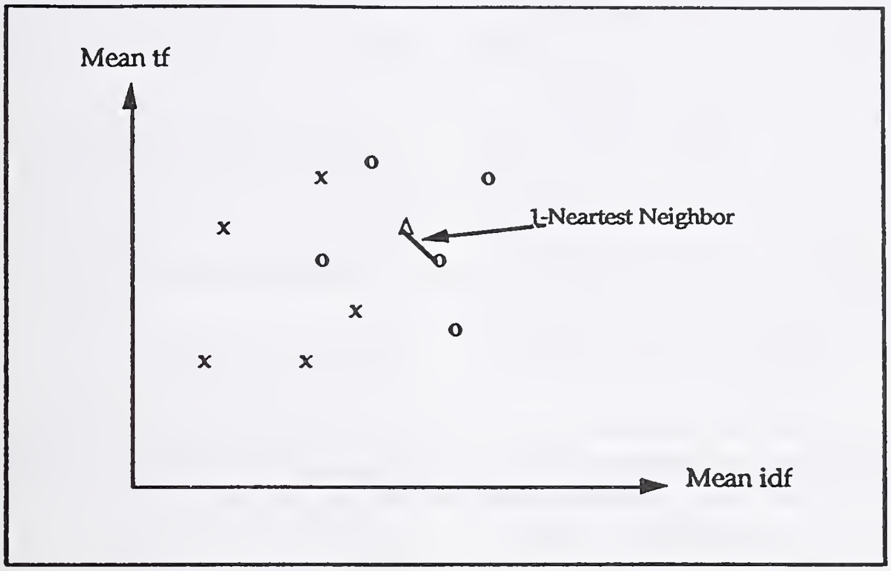


Figure 1: The Nearest Neighbor (1-NN) of a New Request

For example, in Figure 1, an "x" indicates each query for which the best retrieval scheme is A, and "0" each request for which the best search strategy is B. The new query is depicted by "Δ". After computing the Euclidean distance from this new observation to all others, we find that the closest distance is a query noted "0", leading to the prediction that the best retrieval scheme for this new query is the search strategy B. However, the real situation is more complex as shown in Figure 2.

In the k-NN method, if the constant k is defined as one, the selected search technique is simply the nearest neighbor of the new request (see Figure 1). For another value of k (e.g., five in this study), the system selects the retrieval schemes appearing most often in the set of the k closest requests, and ties, if any, are broken using the prior probability (see Section 2.3).

A final implementation isuue should be discussed. Since each query feature is measured with a different scale, we have standardized each feature value by subtracting the estimated mean and dividing by the estimated standard error. Thus the distance between two observations is computed in terms of standard deviation from the sample mean of each feature.

With each classification method, we must deal with various problems. Firstly, adding more features does not necessary lead to a better prediction results. Secondly, some chosen features may reveal little discrimination power between each search scheme. Thirdly, some features may be redundant to others characteristics. Fourthly, in our attempt to learn from the data, we may eventually infer that better features are needed to make reliable predictions. Finally, we implicitly admit that the sample of queries used to built our selection procedure is a representative sample of future requests.

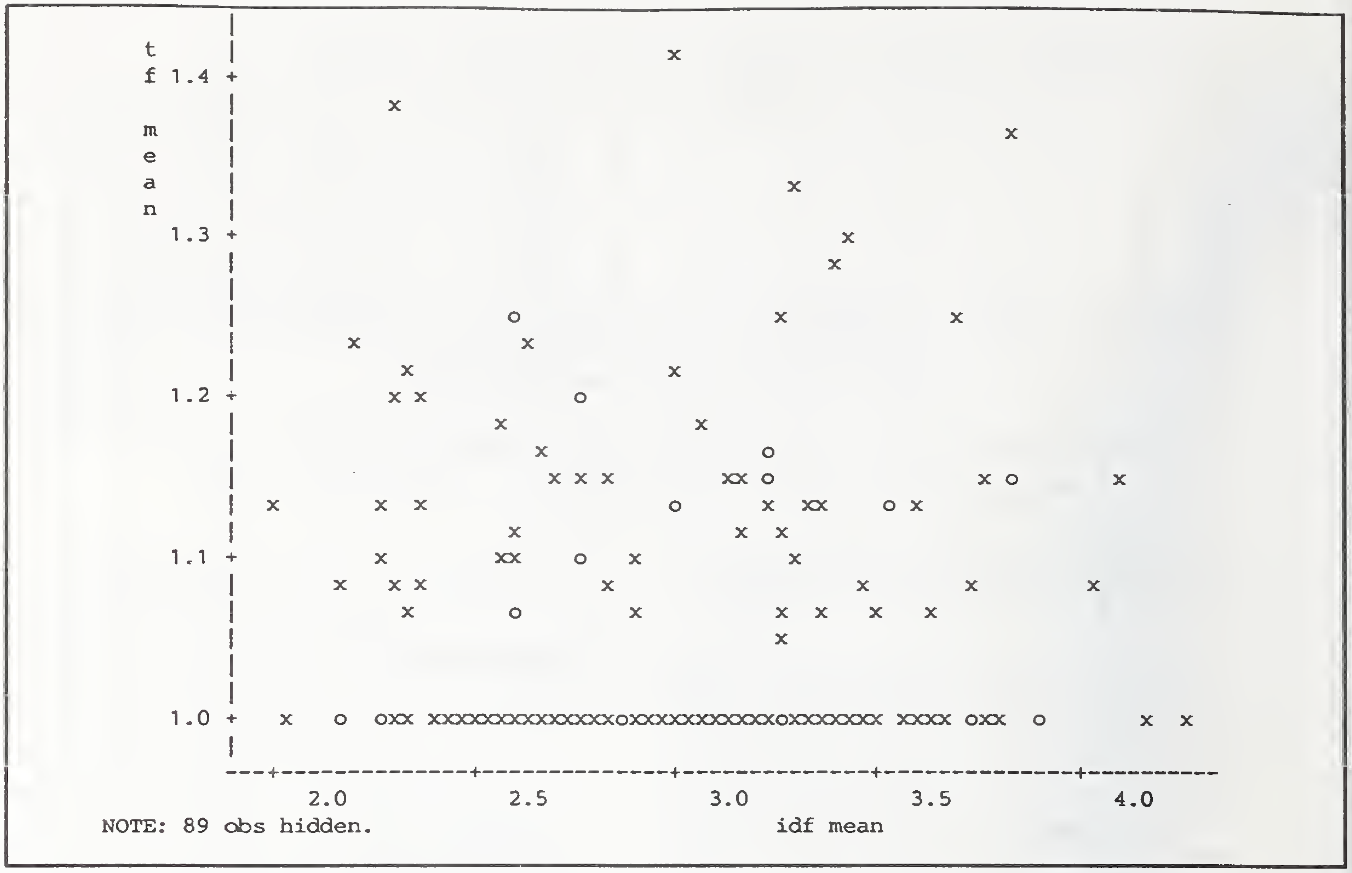


Figure 2: Scattergram of Two Query Characteristics (tf Mean, idf Mean) According to Two Retrieval Schemes

In a related study, McCall & Willett [1986] suggest taking account of the query similarity related to the number of documents within which the search terms occurs (Dice's coefficient) on the one hand, and, on the other hand, the mean similarity between the request and the top 10 retrieved records of each search strategies. Finally, these authors suggest considering the number of relevant items retrieved by each search mechanisms.

> "In view of these results and those obtained by Croft & Thompson [1984], we think it unlikely that automatic selection criteria will be sufficiently discriminating to choose between different search mechanisms in multi-strategy retrieval systems." [McCall & Willett 1986, p. 325]

A direct comparison with this study is not possible because our selection approach must be performed before any search operation. Therefore, in our context, we ignore both the retrieved set and the relevant records obtained by each search scheme.

2.3. The Default Rule

As a first attempt to define a selection rule, we may ignore the query features and based our decision on the prior probabilities. The resulting decision, called the default rule (no data), of our selection procedure is the following: "Select the retrieval scheme having the maximum prior probability". Based on data of Table 5, we may conclude that the OKAPI model returns the best retrieval performance for 132 requests out of 200 (or for 66% of the queries). Thus, the selection of this probabilistic retrieval model is always the response of the default rule.

The evaluation under the label "UNINE3" reflects the result achieved by the OKAPI probabilistic retrieval model for queries from #202 to #250 and represents the baseline approach used for comparisons.

2.3. Official Runs

Considering all the statistical characteristics of each query, our learning scheme based on the 5-NN approach suggests that all queries from #202 to #250 must be processed according to the OKAPI search model. However, this solution is the same as the default rule leading to the conclusion that the picked statistical features of queries cannot be considered as good predictors for an automatic selection of the search strategy.

However, if the computation of the nearest neighbor of a query is based on (1) the number of search keywords, (2) the maximum of the search term frequency, (3) the mean of the search term frequency, (4) the maximum inverse document frequency and (5) the median of the inverse document frequency, another selection will be produced. Thus, the evaluation under the label "UNINE4" reflects the result achieved by our learning procedure based on these five request features (see Table 7).

The selection result depicted in Table 7 is obtained through a learning process based only on the WSJ collection and queries #1 to #200. However, the following question must still be answered: is this selection the optimum for the WSJ collection and for the SJMN corpus when considering topics #202 to #250 (without topic #236)?

When comparing the retrieval effectiveness of the default rule (average precision of 17.97 - 48 queries) with our selection paradigm (17.14), we can see that our approach decreases the overall performance by around 4.6%. Even if this difference cannot be considered as significant, we must still investigate whether this difference appears only on the SJMN collection or also with the WSJ collection (on which the learning process has been done). In Table 8, one can find the optimum retrieval scheme for each query; this optimum selection leads to an average precision of 21.26.

Conclusion

Before all the needed statistical analyses are performed, our preliminary feelings are the following. We think that all statistical query features are not good predictors for an automatic selection procedure. The selection of the appropriate retrieval scheme based on request features is also a difficult problem because the underlying characteristics do not possess very effective powers of discrimination.

Moreover, the optimal combination of various vector-space schemes does not very significantly enhance the particularly high retrieval effectiveness of the OKAPI model. Thus, faced with a good search strategy, an appropriate combination of retrieval schemes seems to be grounded on a radically different indexing strategy (e.g., based on phrase [Bartell et al. 1994]) or on different query formulations (e.g., p-norm and natural language requests [Fox & Shaw 1994]).

However, when considering computation costs, the selection of an appropriate search scheme before any retrieval operation takes place represents an interesting approach.

Query	Search Scheme
204	doc = NNN, query = NNN
233	doc = ANN, query = ANN
205, 238	doc = LTC, query = LTC
others	doc = OKAPI, query = OKAPI

Table 7: Selection of the "Appropriate" Retrieval Scheme (Based on the WSJ Collection)

Query	Search Scheme
	doc = NNN, query = NNN
208, 214, 230, 232, 249	doc = ANN, query = ANN
210, 238	doc = LNC, query = LNC
216, 217, 229, 239, 240	doc = LTC, query = LTC
206, 212	doc = LTC, query = LNC
204, 213, 228, 241	doc = LNC, query = LTC
others	doc = OKAPI, query = OKAPI

Table 8: Selection of the "Optimum" Retrieval Scheme

Acknowledgments

The authors would like to thank Prof. G. Salton and C. Buckley from Cornell University for giving us the opportunity to use the SMART system without which this study could not be conducted. This research was supported by the SNFSR (Swiss National Foundation for Scientific Research) under grants 2100-037345.93 and 20-43'217.95.

References

[Bartell 1994] B. T. Bartell, G. W. Cottrell, R. K. Belew: Automatic Combination of Multiple Ranked Retrieval Systems. Proceedings ACM-SIGIR'94, Dublin (Ireland), June 1994, 173-181.

[Belkin 1993] N. J. Belkin, C. Cool, W. B. Croft, J. P. Callan: The Effect of Multiple Query Representations on Information System Performance. Proceedings ACM-SIGIR'93, Pittsburgh (PA), June 1993, 339-346.

[Belkin 1994] N. J. Belkin, P. Kantor, C. Cool, R. Quatrain: Query Combination and Data Fusion for Information Retrieval. Proceedings TREC'2, Gaithersburg (MD), April 1994, 35-44.

[Buckley 1995] C. Buckley, G. Salton, J. Allen, A. Singhal: Automatic Query Expansion using SMART. Proceedings TREC'3, Gaithersburg (MD), April 1995, 69-80.

[Callan 1995] J. P. Callan, Z. Lu, W. B. Croft: Searching Distributed Collections with Inference Networks. Proceedings ACM-SIGIR'95, Seattle (WA), July 1995, 21-28.

]Croft 1984] W. B. Croft, R. H. Thompson: The Use of Adaptive Mechanisms for Selection of Search Strategies in Document Retrieval Systems. Proceedings ACM-SIGIR'84, Cambridge (UK), July 1984, 95-110.

[Croft 1987] W. B. Croft, R. H. Thompson: I^{3}R: A New Approach to the Design of Document Retrieval Systems. Journal of the American Society for Information Science, 38(6), 1987, 389-404.

[Fox 1988] E. A. Fox, G. L. Nunn, W. C. Lee: Coefficients for Combining Concept Classes in a Collection. Proceedings ACM-SIGIR'88, Grenoble (France), June 1988, 291-307.

[Fox 1993] E. A. Fox, M. P. Koushik, J. Shaw, R. Modlin, D. Rao: Combining Evidence from Multiple Searches. Proceedings TREC'1, Gaithersburg (MD), March 1993, 319-328.

[Fox 1994] E. A. Fox, J. A. Shaw: Combination of Multiple Searches. Proceedings TREC'2, Gaithersburg (MD), March 1994, 243-249.

[Harman 1994] D. Harman (Ed.): Proceedings TREC'2. Gaithersburg (MD), April 94.

[Katzer 1982] J. Katzer, M. J. McGill, J. A. Tessier, W. Frakes, P. DasGupta: A Study of the Overlap among Document Representations. Information Technology: Research & Development, 2, 1982, 261-274.

[Lee 1995] J. H. Lee: Combining Multiple Evidence from Different Properties of Weighting Schemes. Proceedings ACM-SIGIR'95, Seattle (WA), July 1995, 180-188.

[McCall 1986] F. M. McCall, P. Willett: Criteria for the Selection of Search Strategies in Best Match Document Retrieval Systems. International Journal of Man-Machine Studies, 25(3), 1986, 317-326.

[Michie 1994] D. Michie, D. J. Spiegelhalter, C. C. Taylor(Ed.): Machine Learning, Neural and Statistical Classification. Ellis Horwood, New-York (NY), 1994.

[Noreault 1981] T. Noreault, M. J. McGill, M. B. Koll: A Performance Evaluation of Similarity Measures, Document Term Weighting Schemes and Representations in a Boolean Environment. In Information Retrieval Research, R.N Oddy, S. E. Robertson, c. J. van Rijsbergen, P. W. Williams (Eds), Butterworths, London, 1981, 57-76, Symposium Research and Development in Information Retrieval, Cambridge (UK), June 1980.

[van Rijsbergen 1979] C. J. van Rijsbergen: Information Retrieval. Butterworths, 2nd ed., London, 1979.

[Robertson 1995a] S. E. Robertson, S. Walker, M. M. Hancock-Beaulieu: Large Test Collection Experiments on an Operational, Interactive System: Okapi at TREC. Information Processing & Management, 31(3), 1995, 345-360.

[Robertson 1995b] S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, M. Gatford: Okapi at TREC-3. Proceedings TREC'3, Gaithersburg (MD), April 1995, 109-126.

[Savoy 1995] J. Savoy: An Evaluation of Probabilistic Retrieval Models. Technical Report, Faculté de droit et des sciences économiques, Université de Neuchâtel, CR-I-95-04, June 1995, p. 29.

[Saracevic 1988] T. Saracevic, P. Kantor: A Study of Information Seeking and Retrieving. III. Searchers, Searches, Overlap. Journal of the American Society for Information Science, 39(3), 197-216.

[Shaw 1995] J. A. Shaw, E. A. Fox: Combination of multiple searches. Proceedings TREC'3, Gaithersburg (MD), April 1995, 105-108.

[Swanson 1977] D. R. Swanson: Information Retrieval as a Trial and Error Process. Library Quarterly, 47(2), 1977, 128-148.

[Tague-Sutcliffe 1995] J. Tague-Sutcliffe, J. Blustein: A Statistical Analysis of the TREC-3 Data. Proceedings TREC'3, Gaithersburg (MD), April 1995, 385-398.

[Tenopir 1985] C. Tenopir: Full Text Database Retrieval Performance. Online Review, 9(2), 1985, 149-164.

[Thompson 1990] P. Thompson: A Combination of Expert Opinion Approach to Probabilistic Information Retrieval. Information Processing & Management, 26(3), 1990, 371-394.

[Turtle 1991] H. Turtle, W. B. Croft: Evaluation of an Inference Network-Based Retrieval Model. ACM Transactions on Information Systems, 9(3), 1991, 187-222.

[Voorhees 1995] E. M. Voorhees, N. K. Gupta, B. Johnson-Laird: The Collection Fusion Problem. Proceedings TREC'3, Gaithersburg (MD), April 1995, 95-104.

[Voorhees 1995] E. M. Voorhees, N. K. Gupta, B. Johnson-Laird: Learning Collection Fusion Strategies. Proceedings SIGIR'95, Seattle (WA), July 1995, 172-179.

[Weiss 1991] S. M. Weiss, C. A. Kulikowski: Computer Systems That Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems. Morgan Kaufmann, San Mateo (CA), 1991.

TREC-4 Experiments using Drift

Charles L. Viles and James C. French

Department of Computer Science
University of Virginia
Charlottesville, VA 22903
{viles,french}@virginia.edu

Abstract

Drift is a prototype, vector space based, information retrieval system in development at the University of Virginia. The system is designed to do experiments in distributed, dynamic information retrieval. We describe our first experiments using Drift on larger test collections, specifically the Category B subset of the TREC corpus.

1 Introduction

The intelligent application of classic Information Retrieval (IR) methods in today's information environment is pivotal to enabling effective, efficient search of distributed document collections. Recent research results in the IR literature [1, 2, 7, 8, 9] illustrate that IR researchers are recognizing and attacking this problem on a number of fronts.

It seems clear that the dynamic, distributed environment of the present and future Internet requires some changes in the way we look at searching document collections. As identified in [2], several interesting research questions arise:

- Collection Selection / Resource Discovery;
- Efficient Search; and
- Combining Search Results / Collection Fusion.

At the University of Virginia, we are particularly interested in the maintenance of *collection wide information (CWI)* (e.g. the idf and dictionary) when the collection itself is composed of separate, distributed sub-collections and where each sub-collection changes over time. Our essential thesis is that strict maintenance of CWI is costly in the loosely coupled Internet environment and is often unnecessary from an effectiveness standpoint. Our experiments to date have been centered around verifying this thesis as well as identifying how "out-of-date" CWI can be while still maintaining effectiveness commensurate with a central, static archive composed of the current corpus.

As is true for many researchers, our impetus in getting involved with TREC was to gain access to very large document collections. Our early results with small collections [7] were promising but exposed some questions that we felt could only be answered by repeating our experiments on much larger document collections.

In this paper we proceed as follows. In Section 2 we describe our general experimental framework and parameters and their embodiment in a set of software tools we call DRIFT. We also describe our TREC runs in this section. In Section 3 we present preliminary results from the Category B runs. We continue with a discussion of our results and their relation to our previous work. We conclude with lessons learned and a short description of possible future work.

2 Algorithms and Methodology

Before explaining the specifics of our experiments, we first explain the technique we use to model "out-of-date" views of collection statistics. We then outline the specific work we did on the Category B subset of the TREC corpus.

2.1 Background

2.1.1 Notation

The distributed archive is composed of s sites. The j-th document at site i is denoted by D_{ij}. At any site i, there are two collections represented. C_i^l represents the ordered collection of documents physically stored at site i - the "local" collection. The order corresponds to the insertion order of documents at that site. C_i^g represents the collection of documents that has been used to generate site i's version of CWI. We call this version G_i, so $G_i = f(C_i^g)$.

2.1.2 Dissemination of Collection Statistics

Let $prefix(d, C_i^l)$ be the first d-th fraction of C_i^l. The parameter d defines the degree of dissemination of CWI in the archive. At any point in time, site i knows about all of its own documents plus $prefix(d, C_j^l)$ $\forall j \neq i$. That is

$$C_i^g = C_i^l \cup \left(\bigcup_{j \neq i} prefix(d, C_j^l) \right) \tag{1}$$

Note the following about the dissemination parameter

- d varies continuously between 0 and 1.
- When $d = 0$, no dissemination occurs and G_i is derived solely from local holdings.
- When $0 < d < 1$, G_i is derived partly from local holdings and partly from documents held elsewhere.
- When $d = 1$, complete dissemination occurs. Every site has "perfect" knowledge of every other site. So, $G_i = G_j \forall i, j$.

2.1.3 Allocation Model

One of our early findings was that the content-based allocation of documents to sites had an influence on effectiveness [7]. In particular, the more "skew" there was between the content of the collections at each site, the more dissemination was needed to achieve a specified effectiveness level. We found analogous behavior attributable to dynamism [8].

To gain parametric control over this notion of content-skew, we define an affinity probability which attempts to group content-similar documents at the same site. Our approach assumes that documents that are relevant to the same query are similar in content. We assign each query Q, a random home site, $QHome(Q)$. Documents are then assigned to sites based on three pieces of information:

- relevance information,
- $QHome()$,
- the affinity probability a.

If document D is relevant to query Q, then D is assigned to $QHome(Q)$ with probability a, and is assigned at random across all sites with probability $1 - a$. When $a = 0$, documents are assigned at random to sites in the distributed system without regard to their content. As a approaches 1, documents relevant to the same query are increasingly likely to be colocated. When $a = 1$, content-skew is maximal. Of course this methodology is limited to testing environments in which relevance judgements are available.

2.2 Experimental Framework

2.2.1 Drift

The software we use to run our dissemination and allocation experiments is called Drift. Drift is an object-oriented implementation of the Vector Space Model [5] written in C++ and designed specifically to perform experiments in distributed IR.

The Drift architecture is shown in Figure 1A, and several functions are illustrated in Figure 1B-D. In keeping with the object oriented paradigm, the Drift architecture can be viewed as a series of communicating objects. The fundamental objects in Drift are *Documents*, *Collections*, *Searchers*, *Sites*, *DocStreams*, and *CWIs*. There is also a distinguished object called the *Master* that controls the retrieval experiment.

A *Site* maintains a *Collection* (a group of *Document* objects), a *CWI*, and a *Searcher*. The *CWI* maintains the local view of the collection wide information while the *Searcher* maintains a list of term weighting strategies and the machinery to run searches with these strategies.

The *DocStream* object serves as a source of documents to the IR system. In our architecture, documents from the *DocStream* are channeled through the *Master*. The *Master* decides where the document should reside based upon the allocation policy in place and then sends the document to that site. This is not necessarily the insertion mechanism one would expect in an operational distributed IR system, but it is convenient for research purposes. The source for a *DocStream* can be any collection of documents, though in order to perform evaluation experiments, the collection must have an accompanying set of queries and relevance judgements. Document insertion is depicted in Figure 1B.

When searches need to be conducted (Figure 1C), the *Master* broadcasts the query to each *Site* (Step 1). The *Searcher* at the site then runs the search using the local *CWI* (Step 2) and returns the results to the *Master* for collation with results from other sites (Step 3).

For maximal control over the system, dissemination activities (Figure 1D), are initiated by the *Master* by sending a simple "disseminate" message to each *Site*. Each *Site* then sends the documents on its dissemination list (including documents it received from other *Sites*) to its neighbor *Sites*. The contents of this list are determined by the local dissemination policy, which in turn is configurable on a site-by-site basis. We currently fix both dissemination policy and level to the same values at all sites. Documents received from places other than the *Master* are used to update the local *CWI* object but are not actually inserted in the local collection.

Currently, DRIFT does not maintain an inverted index - all searching is done using the query and document vectors directly. In our small collection experiments [7], inverted indexes were unnecessary as we were able to achieve acceptable performance without them. However, with the large size of the TREC corpus, this initially convenient implementation decision severely constrains the scope of our work. We will have to add inverted index based searching in the future.

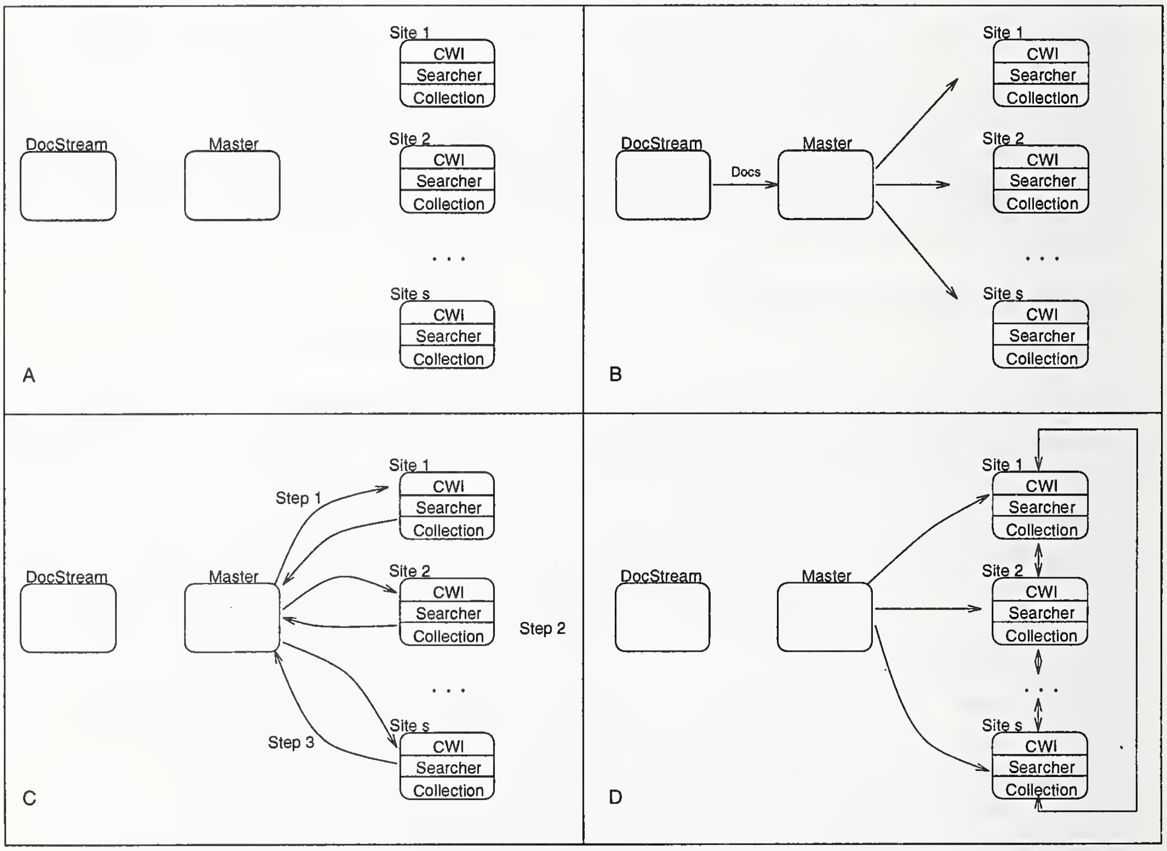


Figure 1: DRIFT architecture. The overall object structure is depicted in (A). Interactions for insertion (B), search (C), and dissemination (D) are also shown.

In Figure 2, we illustrate how document collections are constructed and processed in DRIFT. The raw text of the collection is given as input to the unmodified SMART v11.0 software. SMART removes stopwords, performs word stemming, and produces the dictionary and term frequency based document and topic vectors. We then take these vectors and convert them to a format that is suitable to the access patterns typical of our environment.

Once the collection is constructed, we use the DRIFT format and a parameter file to perform multiple runs for a particular experimental configuration. A configuration consists of values for number of sites, allocation, dissemination, document term weighting, query term weighting and other parameters. Over the life of a suite of experiments, we typically fix all parameters except the allocation and dissemination levels. Multiple runs are necessary because of the stochastic component of allocating documents to sites.

The output from the multiple runs is run through `trec_eval` and a perl program to generate mean and standard deviation estimates for several IR effectiveness measures.

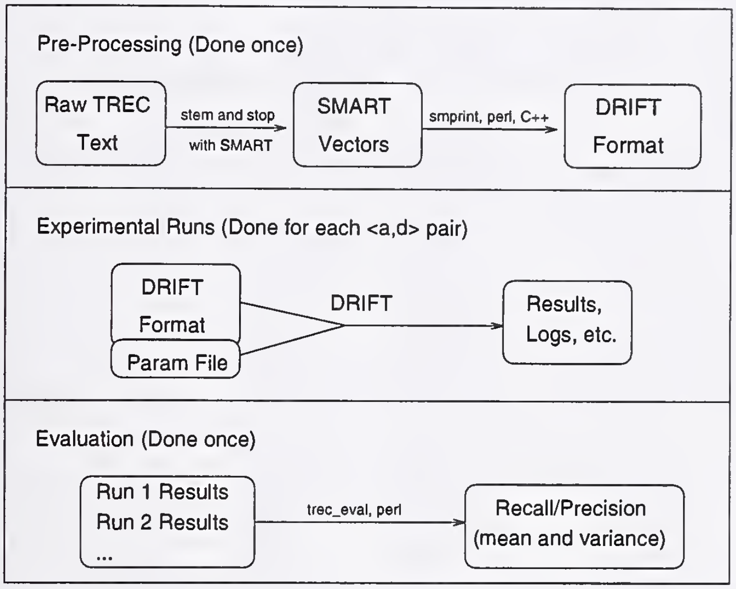


Figure 2: Collection Construction and Processing in DRIFT.

2.2.2 TREC runs

Both document and query construction were completely automatic. Using the notation from Salton and Buckley [4], we used *tfc* weighted documents (i.e. cosine normalized) and *nfx* weighted queries. We made no attempt to determine the best absolute term-weighting strategies for these collections. Collection statistics for both *nfx* and *tfc* weighting are derived from the current site and that site's generally incomplete knowledge of the holdings of other sites.

Our two official runs, *drift1* and *drift2*, represent a centralized system that uses full knowledge of the collection and a distributed system of 20 sites where each site has knowledge of only its local holdings ($d = 0$) and where documents were randomly allocated to the sites ($a = 0$). After discussions with other investigators at TREC-4, we made an additional run using length normalized document term weights rather than cosine normalized weights. This run, *drift3*, is unofficial but we include it to verify that our absolute performance could be improved. The rationale for length normalization is described in Singhal *et al.*[6].

Experiments were run on a Sun SPARCserver 20 with 128 MB memory, a 50 MHz processor, and an attached 2 GB disk. The CPU for the machine was shared with other compute intensive jobs but the disk was dedicated to the TREC work.

3 TREC-4 Results

Construction of the Drift format collection from the raw TREC text took about seven hours. This only needs to be done once per collection. The Drift format collection has about 40% of the space requirement of the raw text. Each run through the Category B data using the test topics took between four and six hours depending upon the processor load.

The results of our official and unofficial TREC-4 runs are in Table 1. Of most interest to us is not absolute performance but the relative performance between the Central (C) and Distributed (D) runs.

	TREC Category B Runs			
	Official			Unofficial
Level	Drift1(C)	Drift2(D)	Δ (%)	Drift3
0.0	0.476	0.469	-1.5	0.680
0.1	0.289	0.290	+0.3	0.449
0.2	0.235	0.213	-9.4	0.308
0.3	0.164	0.153	-6.7	0.241
0.4	0.085	0.086	+1.2	0.171
0.5	0.068	0.066	-2.9	0.140
0.6	0.046	0.046	0.0	0.096
0.7	0.028	0.028	0.0	0.059
0.8	0.005	0.005	0.0	0.036
0.9	0.002	0.002	0.0	0.003
1.0	0.001	0.001	0.0	0.002
Average (non-interp)	0.109	0.105	-3.7	0.172

Table 1: TREC results on the Category B data. Central (C) and Distributed (D) results are shown for two official runs and one unofficial run.

For comparison with TREC-4 we present our results from some small collection experiments in Table 2. The important information in this table is the consistent drop in effectiveness we observed for the random-content, no-dissemination distributed archive when compared to the central archive. This drop was not observed in the TREC runs.

4 Discussion

The choice of the parameters for our two official TREC runs was motivated by our observations on small collections. For two of the four small collections we had used (MED and CACM), we had noticed effectiveness degradation in randomly allocated ($a = 0$) collections with no dissemination. We had posited that the overall *proportion* of documents that was reflected in a site's view of CWI was directly related to effectiveness. It now appears that this hypothesis is incorrect. The TREC-4 results, as well as other experiments [3], indicate that it is not the overall proportion of documents but some minimal *unbiased* sample size that is important in achieving effectiveness commensurate with centralized archives. As shown in Table 3, in the small collections, experiments with 20 sites yielded average sub-collection sizes in the 50–160 document range. Using the Category B data, average sub-collection size was about 8200 documents.

	MED			CACM		
Level	Central	Distributed	Δ (%)	Central	Distributed	Δ (%)
0.0	0.923	0.955	+3.5	0.718	0.670	-6.7
0.1	0.837	0.837	+0.0	0.662	0.586	-11.4
0.2	0.763	0.768	+0.7	0.538	0.484	-10.0
0.3	0.708	0.697	-1.3	0.472	0.409	-13.3
0.4	0.659	0.639	-3.0	0.390	0.338	-13.3
0.5	0.566	0.548	-3.2	0.325	0.276	-17.8
0.6	0.487	0.442	-9.2	0.251	0.217	-13.5
0.7	0.425	0.386	-9.2	0.175	0.142	-18.9
0.8	0.348	0.312	-10.3	0.148	0.115	-22.3
0.9	0.242	0.210	-13.2	0.093	0.070	-24.7
1.0	0.100	0.089	-11.0	0.075	0.053	-29.3
Average (non-interp)	0.537	0.523	- 2.6	0.332	0.294	-11.4

Table 2: Previous results using small collections.

Collection	Size	Sub-collection Size	Avg Precision Δ (%)
MED	1033	52	-2.6
CACM	3204	160	-11.4
TREC Category B	164777	8239	-3.7
WSJ91	42702	2135	-0.4
AP89	84728	4236	+0.7

Table 3: Summary of average subcollection sizes and the difference in non-interpolated average precision between the central site and a 20 site system using no dissemination and random document allocation.

The nature of the TREC tests precluded us from using relevance judgements to skew the allocation of documents to sites as we have done previously [7]. With relevance judgements now available, we can proceed with this work.

In absolute terms, DRIFT was the poorest performer among the Category B participants. Our post-conference failure analysis attributes this to two factors. First, we used sub-optimal term weighting functions for the TREC data. Singhal *et al.*[6] clearly shows that using cosine normalized term weights in the TREC collections yields drastically poorer effectiveness compared to term weights based on length normalization. The second factor was a bug in our query processing that effectively added a random term from another query to every query in the test set. Given the short nature of the queries, the extra term is likely to have degraded effectiveness. The unofficial run (*drift3*) that we ran after fixing these problems yielded effectiveness on par with the other Category B systems.

As is true for many TREC participants, we experienced non-trivial growing pains when modifying our software to handle large collections. The pre-TREC version of DRIFT made massive use of in-core memory and our first revision of the software attempted to address this problem by turning to disk-based structures for the document collection itself. Memory use is still high, as we continue to maintain memory resident versions of collection statistics at each site in the simulated distributed system.

Currently the documents are stored sequentially in vector format on disk. Each query for each run requires a sequential scan through the vectors. To do further experiments on the entire TREC corpus, we really need to implement inverted index structures for the documents. This is especially true since we must do multiple runs through the corpus to identify the variation in effectiveness due to the random nature of the allocation of documents to sites.

5 Summary and Future Plans

The main contribution of the TREC corpus to our work has been to clarify some ambiguous results we had observed using small test collections and to highlight bottlenecks in our retrieval system. It has also given us several directions for future study.

Our short term plans include software modifications to DRIFT in order to improve our throughput on large collections. Once this is done, we can conduct full scale experiments to see how varying content-skew (through the allocation parameter) influences effectiveness. These modifications should also enable us to improve the absolute effectiveness of the system.

In the longer term, we would like to look at the temporally ordered collections (e.g. AP, WSJ, SJMN) to characterize the drift in collection content over time, perhaps deriving CWI from subcollections defined by temporal windows rather than the extant corpus. The applicability of TREC topics for examining this issue is undetermined.

6 Acknowledgements

We thank Donna Harman for encouraging us to re-examine our system with a fresh sense of skepticism. This work was supported in part by NASA GSRP Fellowship NGT-51018 and by NASA CESDIS Grant 5555-25.

References

[1] Nicholas J. Belkin, Paul Kantor, Edward A. Fox, and J. A. Shaw. Combining the Evidence of Multiple Query Representations for Information Retrieval. *Information Processing and Management*, 31(4):431–448, 1995.

[2] James P. Callan, Zhihong Lu, and W. Bruce Croft. Searching Distributed Collections with Inference Networks. In *Proceedings of the 18th International Conference on Research and Development in Information Retrieval (SIGIR95)*, pages 21–29, Seattle, WA, 1995.

[3] James C. French and Charles L. Viles. Ensuring Retrieval Effectiveness in Distributed, Digital Libraries. *To Appear in Journal of Visual Communication and Image Representation*, 1995.

[4] Gerard Salton and Christopher Buckley. Term-Weighting Approaches in Automatic Text Retrieval. *Information Processing and Management*, 24(5):513–523, 1988.

[5] Gerard Salton and Michael McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, 1983.

[6] Amit Singhal, G. Salton, Mandar Mitra, and Chris Buckley. Document Length Normalization. Technical Report CS-95-1529, Department of Computer Science, Cornell University, 1995.

[7] Charles L. Viles and James C. French. Dissemination of Collection Wide Information in a Distributed Information Retrieval System. In *Proceedings of the 18th International Conference on Research and Development in Information Retrieval (SIGIR95)*, pages 12–20, Seattle, WA, July 1995.

[8] Charles L. Viles and James C. French. On the Update of Term Weights in Dynamic Information Retrieval Systems. In *Proceedings of the 4th International Conference on Knowledge and Information Management*, pages 167–174, Baltimore, MD, November 1995.

[9] Ellen Vorhees, Narendra K. Gupta, and Ben Johnson-Laird. Learning Collection Fusion Strategies. In *Proceedings of the 18th International Conference on Research and Development in Information Retrieval (SIGIR95)*, pages 172–179, Seattle, WA, 1995.

APPENDIX A

This appendix contains results for all TREC-4 participants. The first part lists the particpants' organizations and the set of tags for each organization. The next part describes the evaluation techniques and measures used. The final section of the appendix contains the actual results.

ADHOC RUNS

CATEGORY A DATA

Tag	Organization	Query Construction Method
ACQADH	Department of Defense	manual
Brkly10	University of California, Berkeley	manual
Brkly9	University of California, Berkeley	automatic
CLARTF	CLARITECH Corporation/Carnegie Mellon University	manual
CLARTN	CLARITECH Corporation/Carnegie Mellon University	manual
CnQst1	Excalibur Technologies Inc.	manual
CnQst2	Excalibur Technologies Inc.	manual
CrnlAE	Cornell University	automatic
CrnlAL	Cornell University	automatic
DCU951	Dublin City University	manual
DCU952	Dublin City University	manual
INQ201	University of Massachusetts, Amherst	automatic
INQ202	University of Massachusetts, Amherst	manual
INTXT2	InText Systems	manual
citri1	CITRI, Royal Melbourne Institute of Technology	automatic
citri2	CITRI, Royal Melbourne Institute of Technology	automatic
citya1	City University, London	automatic
citym1	City University, London	manual
fsclt1	FS Consulting	manual
fsclt2	FS Consulting	manual
gmu1	George Mason University	manual
gmu2	George Mason University	automatic
issah1	National University of Singapore	automatic
issah2	National University of Singapore	manual
nyuge3	GE/New York University	automatic
nyuge4	GE/New York University	manual
padreA	Australian National University	automatic
padreZ	Australian National University	manual
pircs1	Queens College, CUNY	automatic
pircs2	Queens College, CUNY	manual
siems1	Siemens Corporate Research Inc.	automatic
uwgcl1	University of Waterloo	manual
virtu4	NEC Corporation automatic	automatic

CATEGORY B DATA

Tag	Organization	Query Construction Method
KU1	University of Kansas	automatic
UniNE3	Universite de Neuchatel	automatic
UniNE4	Universite de Neuchatel	automatic
drift1	University of Virginia	automatic
drift2	University of Virginia	automatic
glair1	University of Glasgow	automatic

ROUTING RUNS

CATEGORY A DATA

Tag	Organization	Query Construction Method
ACQROU	Department of Defense	manual
Brkly11	University of California, Berkeley	automatic
Brkly12	University of California, Berkeley	automatic
CrnlRE	Cornell University	automatic
CrnlRL	Cornell University	automatic
HNC11	HNC, Inc.	automatic
HNC21	HNC, Inc.	automatic
INQ203	University of Massachusetts, Amherst	automatic
INQ204	University of Massachusetts, Amherst	automatic
ORAdL1	Oracle Corporation	automatic
ORAdL2	Oracle Corporation	automatic
UCF100	University of Central Florida	manual
cityr1	City University, London	automatic
cityr2	City University, London	automatic
itidp1	Information Technology Institute, Singapore	automatic
itidp2	Information Technology Institute, Singapore	automatic
losPA2	Logicon Operating Systems	automatic
losPA3	Logicon Operating Systems	automatic
nyuge1	GE/New York University	automatic
nyuge2	GE/New York University	automatic
pircsC	Queens College, CUNY	automatic
pircsL	Queens College, CUNY	automatic
uwgcl1	University of Waterloo	manual
virtu3	NEC Corporation	automatic
xerox1	Xerox Palo Alto Research Center	automatic
xerox2	Xerox Palo Alto Research Center	automatic

CATEGORY B DATA

Tag	Organization	Query Construction Method
ORAdL1	Oracle Corporation	automatic
ORAdL2	Oracle Corporation	automatic

TRACKS

DATABASE MERGING CATEGORY A ADHOC

Tag	Organization	Query Construction Method
inq205	University of Massachusets, Amherst	automatic
inq206	University of Massachusets, Amherst	automatic
inq207	University of Massachusets, Amherst	automatic
inq208	University of Massachusets, Amherst	automatic
inq209	University of Massachusets, Amherst	automatic
padreW	Australian National University	automatic
siems2	Siemens Corporate Research Inc.	automatic
siems3	Siemens Corporate Research Inc.	automatic

SPANISH

Tag	Organization	Query Construction Method
ACQSPA	Department of Defense	manual
BrklySP3	University of California, Berkeley	automatic
BrklySP4	University of California, Berkeley	manual
CrnlSE	Cornell University	automatic
DCUSP0	Dublin City University	manual
SIN010	University of Massachusetts, Amherst	automatic
SIN011	University of Massachusetts, Amherst	automatic
UCFSP1	University of Central Florida	manual
UCFSP2	University of Central Florida	manual
citri-sp1	CITRI, Royal Melbourne Institute of Technology	automatic
citri-sp2	CITRI, Royal Melbourne Institute of Technology	automatic
gmuauto	George Mason University	automatic
gmuman	George Mason University	manual
nmsu0	New Mexico State University	automatic
nmsu1	New Mexico State University	manual
nmsu2	New Mexico State University	manual
nmsu3	New Mexico State University	manual
nmsu4	New Mexico State University	manual
nmsu5	New Mexico State University	manual
xerox-sp1	Xerox Palo Alto Research Center	automatic
xerox-sp2	Xerox Palo Alto Research Center	automatic

CONFUSION CATEGORY B ADHOC

Tag	Organization	Query Construction Method
ACQC10	Department of Defense	manual
ACQC20	Department of Defense	manual
ACQUNC	Department of Defense	manual
CrnlB	Cornell University	automatic
CrnlBc10	Cornell University	automatic
gmuc0	George Mason University	automatic
gmuc10	George Mason University	automatic
rutfum	Rutgers University	manual
rutfuv	Rutgers University	manual
rutscn20	Rutgers University	manual

FILTERING CATEGORY A ROUTING

Tag	Organization	Query Construction Method
ACQHPr	Department of Defense	manual
ACQHRe	Department of Defense	manual
ACQMID	Department of Defense	manual
HNC11	HNC, Inc.	manual
HNC21	HNC, Inc.	manual
HNC31	HNC, Inc.	manual
filt1	Queens College, CUNY	manual
filt2	Queens College, CUNY	manual
filt3	Queens College, CUNY	manual
pircs1	Queens College, CUNY	manual
pircs2	Queens College, CUNY	manual
pircs3	Queens College, CUNY	manual
Xerox25	Xerox Palo Alto Research Center	manual
Xerox50	Xerox Palo Alto Research Center	manual
Xerox75	Xerox Palo Alto Research Center	manual

INTERACTIVE CATEGORY A ADHOC

Tag	Organization	Query Construction Method
CrnlAE	Cornell University	automatic
CrnlI1	Cornell University	manual
CrnlI2	Cornell University	manual
LNBOOL	Lexis-Nexis	manual
cityi1	City University, London	manual
ruibsl	Rutgers University	automatic
ruint1	Rutgers University	manual
ucla	University of California, Los Angeles	manual

Evaluation Techniques and Measures

Categories

The results following this section are organized according to the task accomplished by the run: adhoc, routing, or a track task (confusion, database merging, filtering, interactive, or Spanish).

I. Adhoc.

Retrieval using an "adhoc" topic such as a researcher might use in a library environment. In TREC this implies that the input topic has no training material such as relevance judgments to aid in the construction of the input query.

A. Category A.

Systems running TREC topics against all documents from Disks 2 and 3 of the Tipster Collection.

B. Category B.

Systems running TREC topics against the newspaper documents — the *Wall Street Journal* (WSJ) and the *San Jose Mercury News* (SJMN) — on Disks 2 and 3 of Tipster Collection. (Intended for new groups, allowing them to scale their systems to handle large collections.)

II. Routing.

Retrieval using a "routing" query such as a profile to filter some incoming document stream. In TREC this implies that the input topic has training material, including relevance judgments against the training documents, to use in constructing the input query or profile. This query is then used against new documents (the test documents).

A. Category A.

Systems running TREC topics against approximately 750 megabytes of *Federal Register* and computer documents. The computer documents consisted of documents from computer-related electronic newsgroups as well as the Ziff documents on Tipster Disk 3. The *Federal Register* from 1994 completed the routing data.

B. Category B.

Systems running TREC topics against documents from the *Federal Register* 1994 and Ziff documents on Tipster Disk 3. (Intended for new groups, allowing them to scale their systems to handle large collections.)

Evaluation Measures

I. Recall.

A measure of the ability of a system to present all relevant items,

$$\text{recall} = \frac{\text{number of relevant items retrieved}}{\text{number of relevant items in collection}}.$$

II. Precision.

A measure of the ability of a system to present only relevant items,

$$\text{precision} = \frac{\text{number of relevant items retrieved}}{\text{total number of items retrieved}}.$$

III. Fallout.

A measure of the ability of a system to filter out non-relevant items,

$$\text{fallout} = \frac{\text{number of nonrelevant items retrieved}}{\text{total number of nonrelevant items in collection}}.$$

System Results Description

Each page contains all the results for one system comprised of a header, 4 tables, and 3 graphs described as follows:

Header

- The header contains the task and organization name, where task is either adhoc or routing and the organization is the organization name producing the run described on the page.

Tables

Tables are generated by *trec_eval* courtesy of Chris Buckley using the SMART methodology as defined in Salton and McGill [1].

I. "Summary Statistics" Table.
Table 1 is a sample "Summary Statistics" Table.

TABLE 1
Sample "Summary Statistics" Table.

Summary Statistics	
Run Number	INQ203–category A, automatic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	6576
Rel_ret:	5612

A. Run Number.
An identifier for a system of the form: *Tag - Category, Query Construction Method*, where *Tag* is the id of the run provided by the participant, *Category* is either Category A or Category B (full documents or a subset of full documents), and *Query Construction Method* is either automatic, manual, or interactive.

B. Number of Topics.
Number of topics searched in this run (normally 49 for adhoc, 50 for routing).

C. Total number of documents over all topics (the number of topics shown in B).

i. Retrieved.
Number of documents retrieved and sent in. This is normally 50,000 (50 topics × 1000 documents), but could be less.

ii. Relevant.

Total possible relevant documents within a given task and category. Table 2 contains the possible relevant documents for TREC-4.

TABLE 2

Possible relevant documents within a given task and category for TREC-4.

Category	Adhoc	Routing
A	6501	6576
B	2480	6099

iii. Rel_ret.

Total number of relevant documents returned by a run for all the topics, i.e. the Number of Topics shown on the second line of the Summary Table.

II. "Recall Level Precision Averages" Table.

Table 3 is a sample "Recall Level Precision Averages" Table.

TABLE 3

Sample "Recall Level Precision Averages" Table.

Recall Level Precision Averages	
Recall	Precision
0.00	0.7759
0.10	0.5910
0.20	0.5244
0.30	0.4735
0.40	0.4213
0.50	0.3595
0.60	0.3031
0.70	0.2365
0.80	0.1605
0.90	0.0664
1.00	0.0031
Average precision over all relevant docs	
non-interpolated	0.3419

A. Precision at 11 recall cutoff values.

Each recall-precision average is computed by summing the precisions at the specified recall cutoff value (denoted by $\sum P_R$ where P_R is the precision at recall cutoff value R) and then dividing by the number of topics, for TREC normally $NUM = 50$.

$$\frac{\sum_{i=1}^{NUM} P_R}{NUM} \qquad R = \{0.0, 0.1, 0.2, 0.3, \ldots, 1.0\}$$

- Interpolating recall-precision.
  In order to graphically show precision at various recall averages, interpolation must be used. The interpolated precision at a recall cutoff *R* is defined to be the maximum precision at all points ≤ R.

  For example, if there are only 3 relevant documents retrieved, and these are retrieved at ranks 4, 9, and 20, then the exact recall points are 0.33, 0.67, and 1.0. Interpolated precisions are computed using the "true" recall values (precision 0.25 at recall 0.33, precision 0.22 at recall 0.67, and precision 0.15 at recall 1.0, respectively) and mapping them to the 11 recall cutoff values using the above rule. Therefore, the precisions at recall points 0.0, 0.1, 0.2, 0.3 are 0.25, the precision at recall points 0.4, 0.5 0.6, are 0.22 and precision at recall points 0.7, 0.8, 0.9, 1.0 are 0.15. Note that theoretically precision is not defined at a recall of 0.0, however this interpolation rule allows values to be determined.

B. Average precision over all relevant documents, non-interpolated.
This measure is not an average of the above cutoff values, but an average calculated after each relevant returned document.

Consider a system returning 10 documents and four of the 10 documents are relevant. The rankings of the four documents are 1, 2, 4, 7 giving precisions of 1, 1, 0.75, and 0.57, respectively. By averaging the 4 precisions the single value measure of average precision over all relevant documents is 0.83.

III. "Document Level Averages" Table.
Table 4 is a sample "Document Level Averages" Table.

TABLE 4
Sample "Document Level Averages" Table.

Document Level Averages	
	Precision
At 5 docs	0.5760
At 10 docs	0.5540
At 15 docs	0.5480
At 20 docs	0.5370
At 30 docs	0.5187
At 100 docs	0.4168
At 200 docs	0.3413
At 500 docs	0.2254
At 1000 docs	0.1453
R−Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3890

A. Precision at 9 document cutoff values.
Each document precision average is computed by summing the (50) precisions at the specified document cutoff value divided by the number of topics (50).

B. R-Precision is the precision after R documents (whether relevant or non-relevant) have been retrieved, where R is the number of relevant documents for a topic. R-Precisions are computed, one for each query (50), and then they are averaged.

Suppose a topic with 50 relevant documents was run returning 200 documents. In the top 50 documents returned, 17 of them are relevant. Then the R-Precision is $\frac{17}{50}$ or 0.34.

IV. "Recall Fallout Averages" Table.
Table 5 is a sample "Recall Fallout Averages" Table.

TABLE 5
Sample "Recall Fallout Averages" Table.

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00003
0.20	0.00008
0.30	0.00014
0.40	0.00023
0.50	0.00037
0.60	0.00058
0.70	0.00094
0.80	0.00175
0.90	0.00528
1.00	0.13424

A. Fallout at 11 recall cutoff values.
Each recall-fallout average is computed by summing the fallouts at the specified recall cutoff value (denoted by $\sum F_R$ where F_R is the fallout at recall cutoff value R) and then divided by the number of topics ($NUM = 50$).

$$\frac{\sum_{i=1}^{NUM} F_R}{NUM} \qquad R = \{0.0, 0.1, 0.2, 0.3, \ldots, 1.0\}$$

- Interpolating recall-fallout.
The recall at X non relevant documents is used for the recall at $X - 1$ non relevant documents. When fallout is not exactly defined at R, the fallout of $X - 1$ is used.

Tabular Interpretation

I. Recall Level Precision Averages.

A. Precision at 11 cutoff values.
This table allows comparisons of systems.

B. Average precision over all relevant documents.
This is a single valued number which reflects the performance over all relevant documents. It is intended to reward those systems retrieving relevant documents quickly (highly ranked).

II. Document Level Averages.

A. Precision at 9 document cutoff values.
Document level reflects actual measured system performance as a user might see it. However, the averages computed using document-level measures are difficult to compare.

B. R-Precision.
It is a measure that is intended to de-emphasize the exact ranking of the documents. As such, it is especially valuable for examining routing systems, where the real-life criteria is whether a document is given to a user, rather than at what rank the document would be. R-Precision is particularly useful in TREC where there are large numbers of relevant documents.

III. Recall Fallout Averages.
Fallout is a parallel measure to recall for measuring the nonrelevant documents. An effective retrieval system will exhibit maximum recall and minimal fallout.

Graphs

I. Recall-Precision Graph.
Figure 1 is a sample Recall-Precision Graph.

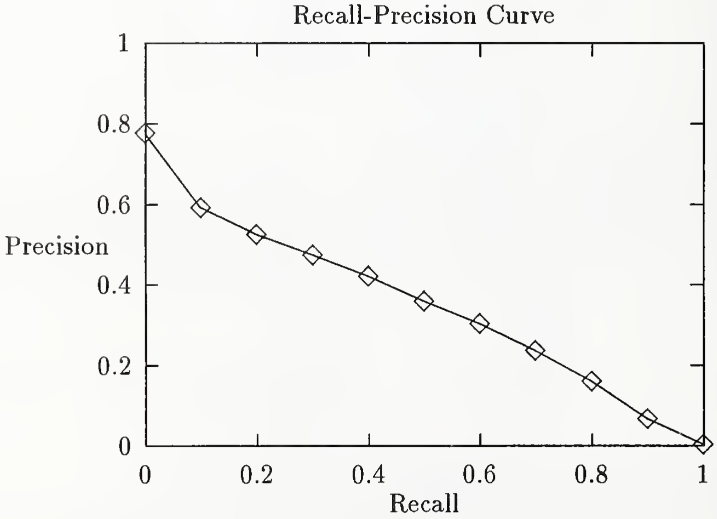


FIG. 1. *Sample Recall-Precision Graph.*

The Recall-Precision Graph is created using the 11 cutoff values from the Recall Level Precision Averages. This graph is useful for comparing systems. The graphs of different systems can be superimposed on the same graph to determine which system is superior. Curves closest to the upper right-hand corner of the graph (where recall and precision are maximized) indicates the best performance. Comparisons are best made in three different recall ranges: 0 to 0.2, 0.2 - 0.8, and 0.8 to 1. These ranges characterize systems as high precision, middle recall, and high recall, respectively.

II. Average Precision Histogram.

Figure 2 is a sample Average Precision Histogram.

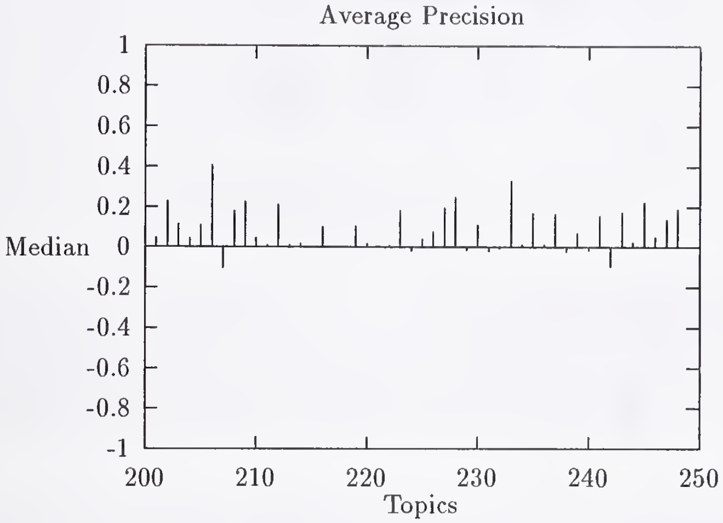


FIG. 2. *Sample Average Precision Histogram.*

The Average Precision Histogram measures the precision of a system on each topic against the median precision of all systems on that topic.

III. Recall-Log Fallout Curve.

Figure 3 is a sample Recall-Log Fallout Curve.

The Recall-Log Fallout Graph is generated using the 11 cutoff values from the Recall Fallout Averages (exactly like the Recall-Precision Graph). Curves closest to the lower left-hand corner of the graph (where recall is maximized and fallout is minimized) indicates the best performance. Fallout is graphed using a logarithmic scale for viewing purposes.

Graphical Interpretation

I. Recall-Precision Graph.

This graph is the most common used to compare information retrieval systems. Typically these graphs slope downward from left to right, enforcing the notion that as more relevant documents are retrieved (recall increases) then the more non relevant documents are retrieved (precision decreases).

II. Average Precision Histogram.

This graph is intended to give insight into the performance of individual systems and the types of topics they handle well.

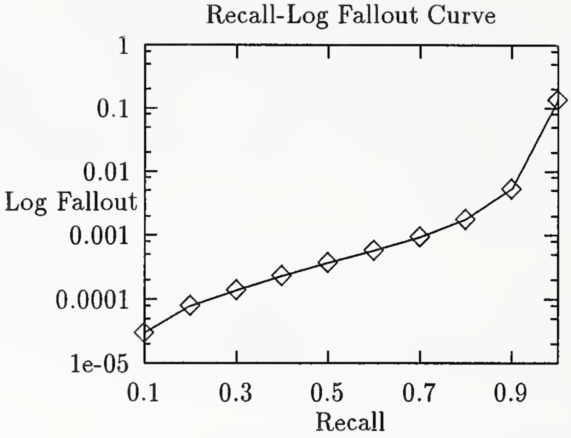


FIG. 3. *Sample Recall-Log Fallout Curve.*

III. Recall-Log Fallout Curve.

This graph illustrates the same concept as the recall-precision graph, except the fallout measure is used. As systems retrieve all relevant documents (recall increases) the number of non-relevant documents returned increases (fallout increases).

References

[1] G. SALTON AND M. MCGILL, *Introduction to Modern Information Retrieval*, McGraw-Hill, New York, first ed., 1983.

Summary Statistics	
Run Number	ACQADH–category A, manual
Number of Topics	49

Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	2713

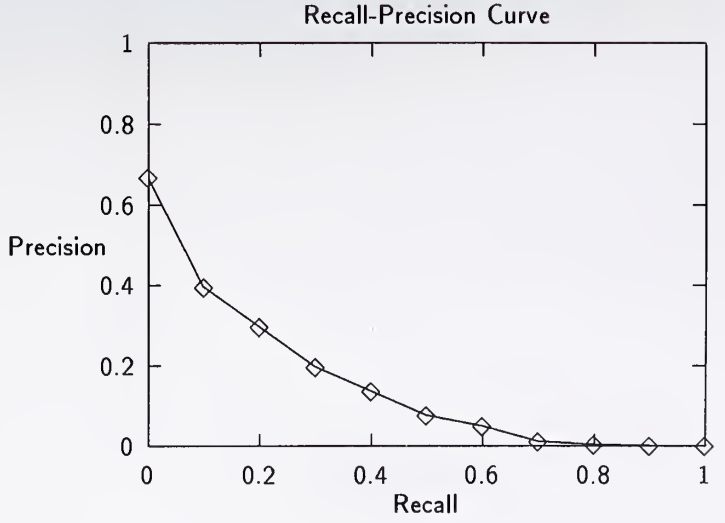
Recall Level Precision Averages	
Recall	Precision
0.00	0.6678
0.10	0.3966
0.20	0.2970
0.30	0.1965
0.40	0.1372
0.50	0.0769
0.60	0.0495
0.70	0.0117
0.80	0.0025
0.90	0.0000
1.00	0.0000

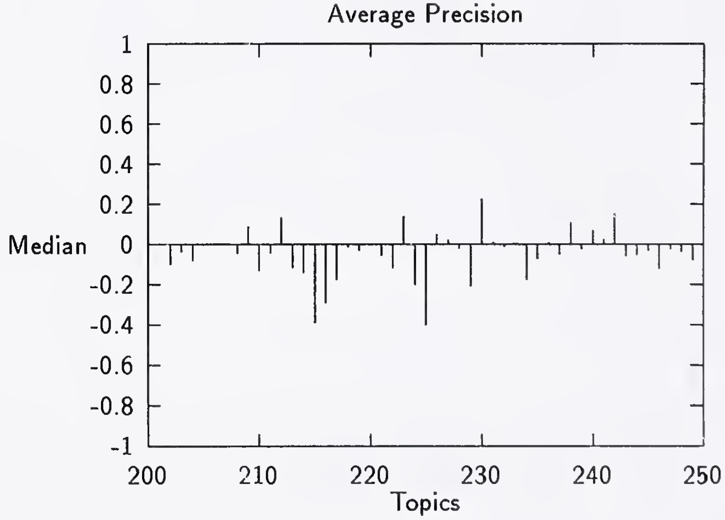
Average precision over all relevant docs	
non-interpolated	0.1385

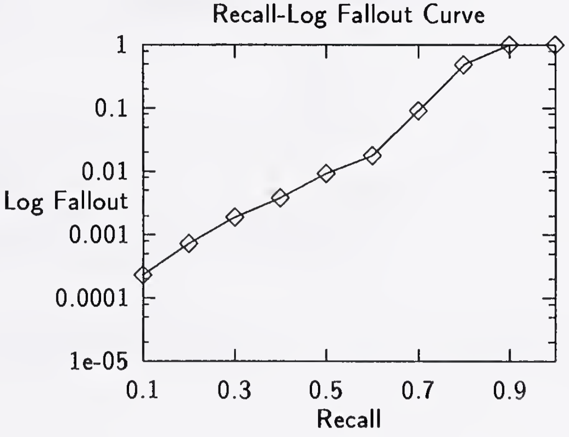
Document Level Averages	
	Precision
At 5 docs	0.4082
At 10 docs	0.3796
At 15 docs	0.3469
At 20 docs	0.3255
At 30 docs	0.2932
At 100 docs	0.1992
At 200 docs	0.1402
At 500 docs	0.0852
At 1000 docs	0.0554

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2023

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00023
0.20	0.00072
0.30	0.00188
0.40	0.00385
0.50	0.00919
0.60	0.01764
0.70	0.09054
0.80	0.48877
0.90	1.00000
1.00	1.00000


Recall-Precision Curve


Average Precision

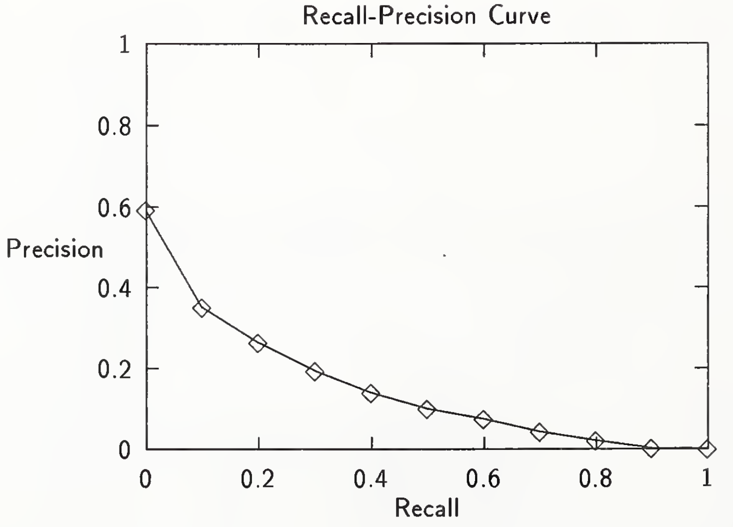

Recall-Log Fallout Curve

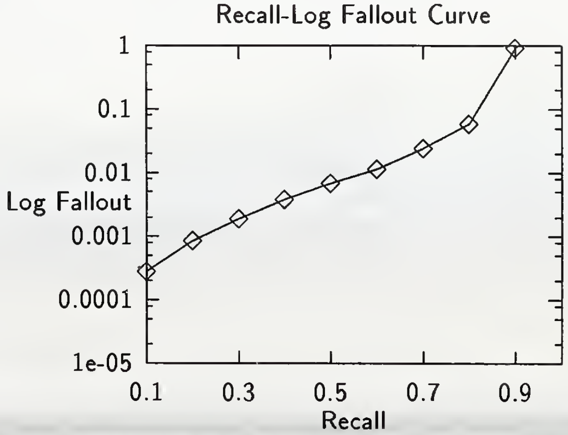
Summary Statistics	
Run Number	Brkly9–category A, automatic
Number of Topics	49
Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	2232

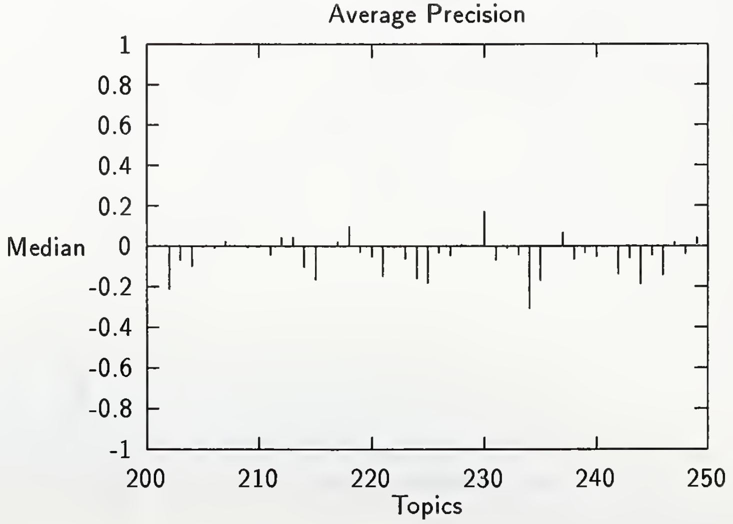
Recall Level Precision Averages	
Recall	Precision
0.00	0.5913
0.10	0.3513
0.20	0.2639
0.30	0.1946
0.40	0.1398
0.50	0.1008
0.60	0.0740
0.70	0.0430
0.80	0.0208
0.90	0.0015
1.00	0.0003
Average precision over all relevant docs	
non-interpolated	0.1388

Document Level Averages	
	Precision
At 5 docs	0.4041
At 10 docs	0.3816
At 15 docs	0.3374
At 20 docs	0.3163
At 30 docs	0.2694
At 100 docs	0.1663
At 200 docs	0.1169
At 500 docs	0.0687
At 1000 docs	0.0456
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1873

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00028
0.20	0.00085
0.30	0.00190
0.40	0.00377
0.50	0.00683
0.60	0.01150
0.70	0.02386
0.80	0.05767
0.90	0.91736
1.00	5.10258







Summary Statistics	
Run Number	Brkly10–category A, manual
Number of Topics	49

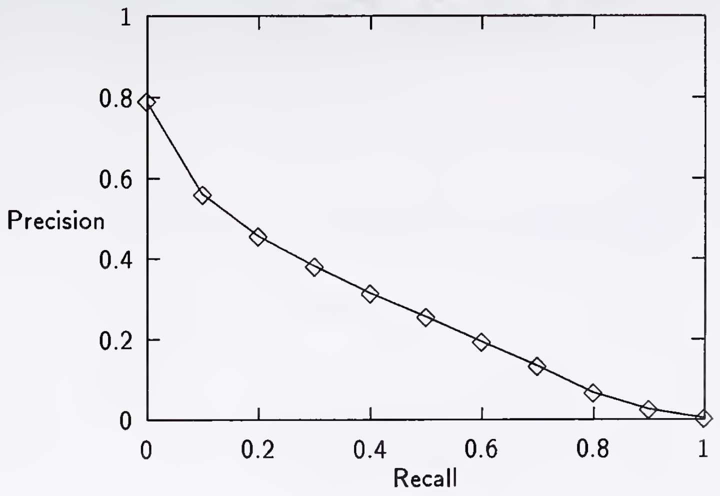
Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	4032

Recall Level Precision Averages	
Recall	Precision
0.00	0.7902
0.10	0.5599
0.20	0.4562
0.30	0.3818
0.40	0.3154
0.50	0.2564
0.60	0.1942
0.70	0.1342
0.80	0.0668
0.90	0.0254
1.00	0.0037
Average precision over all relevant docs	
non-interpolated	0.2660

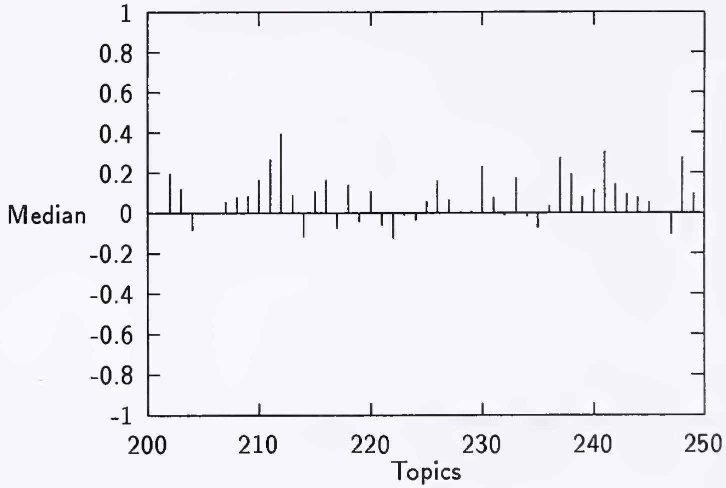
Document Level Averages	
	Precision
At 5 docs	0.5755
At 10 docs	0.5306
At 15 docs	0.5156
At 20 docs	0.4827
At 30 docs	0.4469
At 100 docs	0.3000
At 200 docs	0.2212
At 500 docs	0.1324
At 1000 docs	0.0823
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3219

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00012
0.20	0.00037
0.30	0.00074
0.40	0.00133
0.50	0.00222
0.60	0.00381
0.70	0.00692
0.80	0.01711
0.90	0.05288
1.00	0.41232

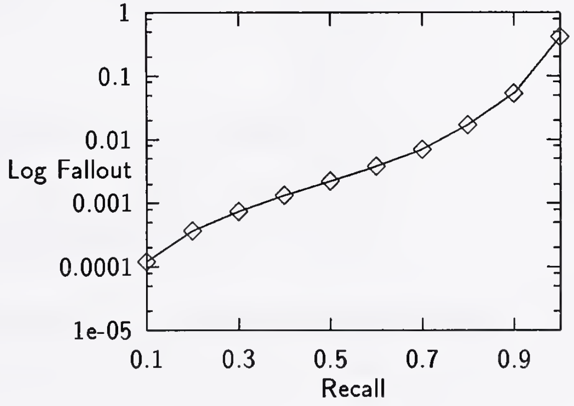
Recall-Precision Curve



Average Precision



Recall-Log Fallout Curve



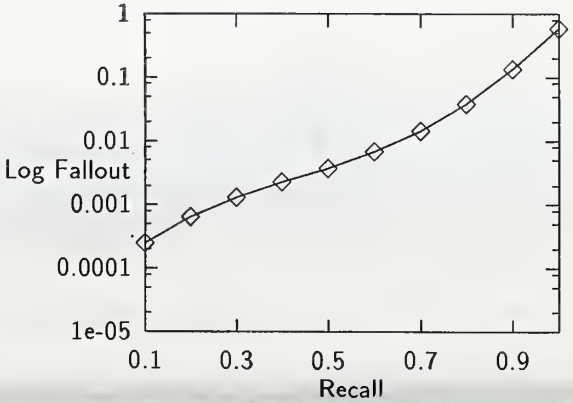
Summary Statistics	
Run Number	citri1–category A, automatic
Number of Topics	49
Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	3067

Recall Level Precision Averages	
Recall	Precision
0.00	0.5946
0.10	0.3839
0.20	0.3201
0.30	0.2607
0.40	0.2134
0.50	0.1711
0.60	0.1188
0.70	0.0699
0.80	0.0308
0.90	0.0102
1.00	0.0026
Average precision over all relevant docs	
non-interpolated	0.1782

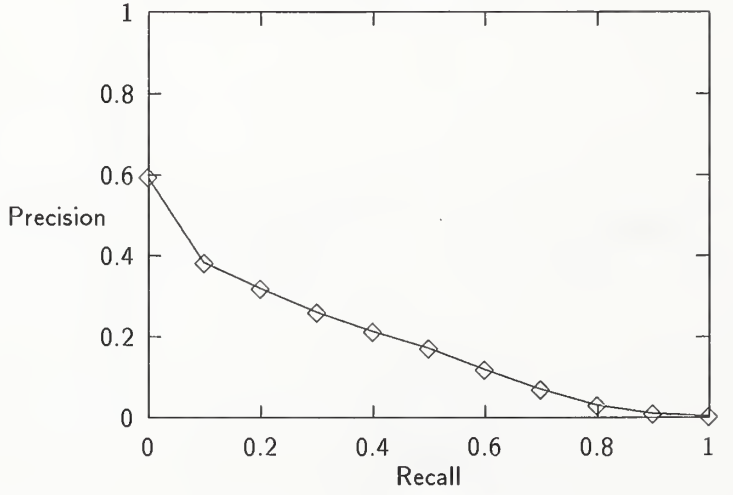
Document Level Averages	
	Precision
At 5 docs	0.3796
At 10 docs	0.3755
At 15 docs	0.3415
At 20 docs	0.3173
At 30 docs	0.2993
At 100 docs	0.2073
At 200 docs	0.1560
At 500 docs	0.0967
At 1000 docs	0.0626
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2322

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00025
0.20	0.00065
0.30	0.00130
0.40	0.00226
0.50	0.00371
0.60	0.00681
0.70	0.01426
0.80	0.03855
0.90	0.13373
1.00	0.58740

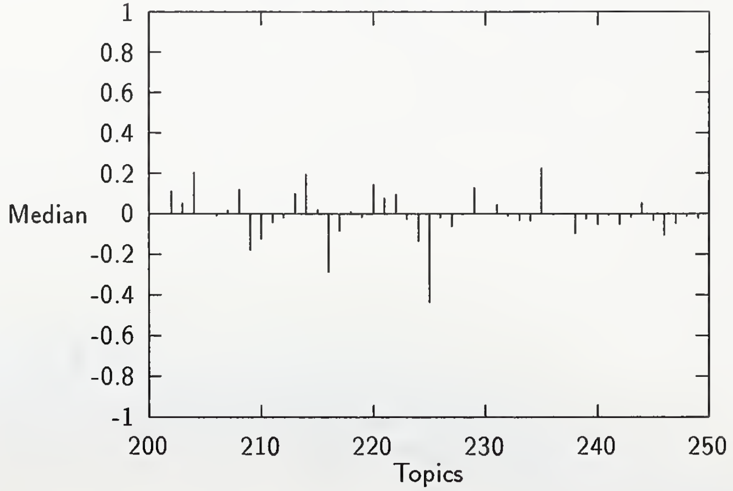
Recall-Log Fallout Curve



Recall-Precision Curve



Average Precision



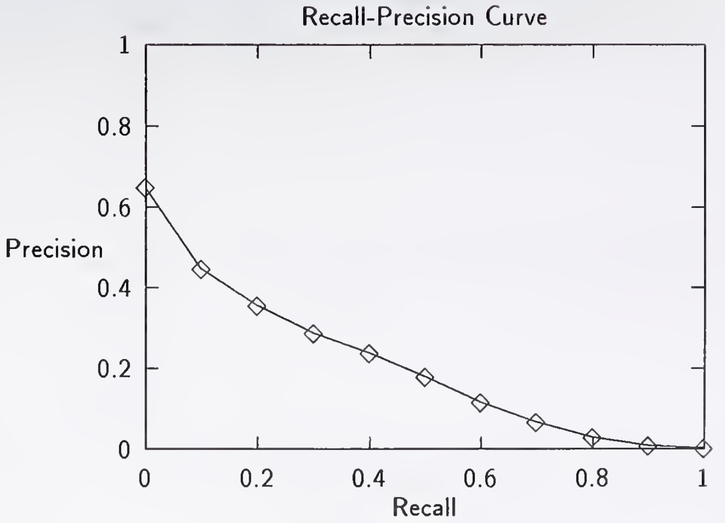
Summary Statistics	
Run Number	citri2–category A, automatic
Number of Topics	49

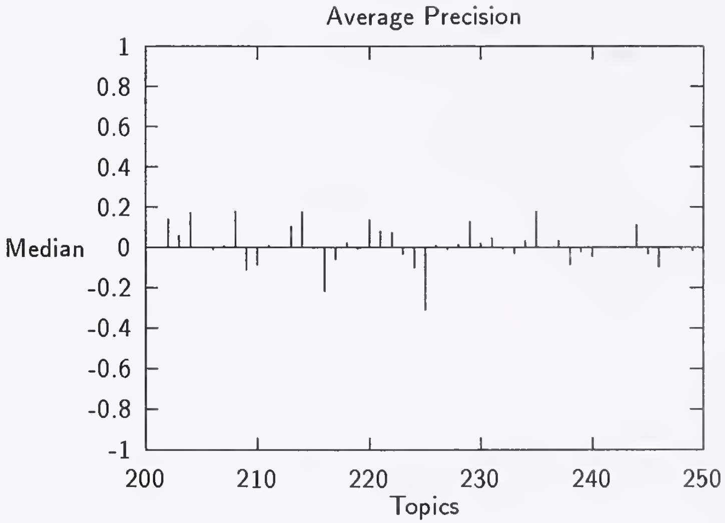
Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	3127

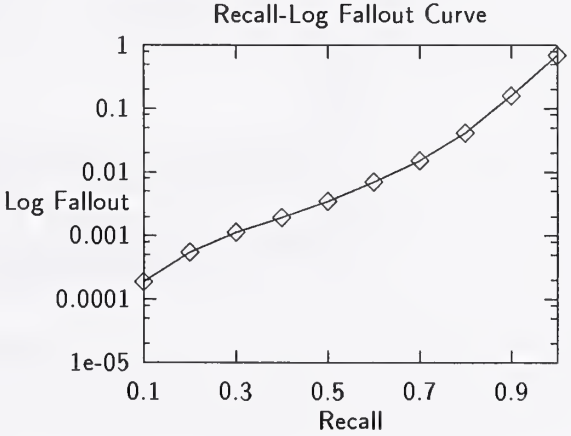
Recall Level Precision Averages	
Recall	Precision
0.00	0.6480
0.10	0.4469
0.20	0.3560
0.30	0.2871
0.40	0.2383
0.50	0.1792
0.60	0.1158
0.70	0.0663
0.80	0.0288
0.90	0.0087
1.00	0.0022
Average precision over all relevant docs	
non-interpolated	0.1956

Document Level Averages	
	Precision
At 5 docs	0.4776
At 10 docs	0.4367
At 15 docs	0.3946
At 20 docs	0.3724
At 30 docs	0.3361
At 100 docs	0.2310
At 200 docs	0.1716
At 500 docs	0.1015
At 1000 docs	0.0638
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2581

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00019
0.20	0.00055
0.30	0.00114
0.40	0.00196
0.50	0.00351
0.60	0.00702
0.70	0.01509
0.80	0.04131
0.90	0.15703
1.00	0.69448







Summary Statistics	
Run Number	citya1–category A, automatic
Number of Topics	49

Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	3807

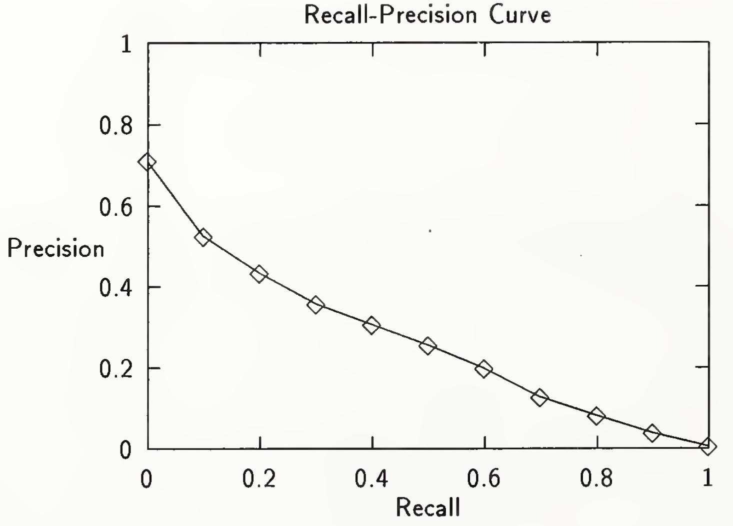
Recall Level Precision Averages	
Recall	Precision
0.00	0.7106
0.10	0.5242
0.20	0.4341
0.30	0.3583
0.40	0.3068
0.50	0.2555
0.60	0.1986
0.70	0.1272
0.80	0.0806
0.90	0.0376
1.00	0.0055

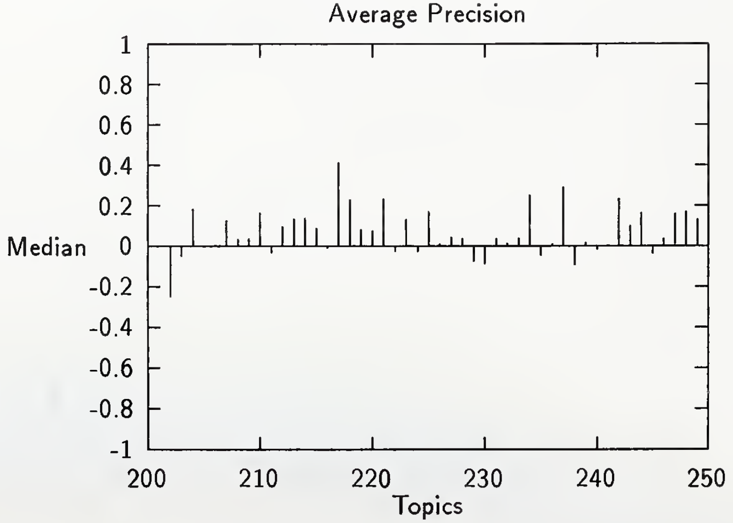
Average precision over all relevant docs	
non-interpolated	0.2568

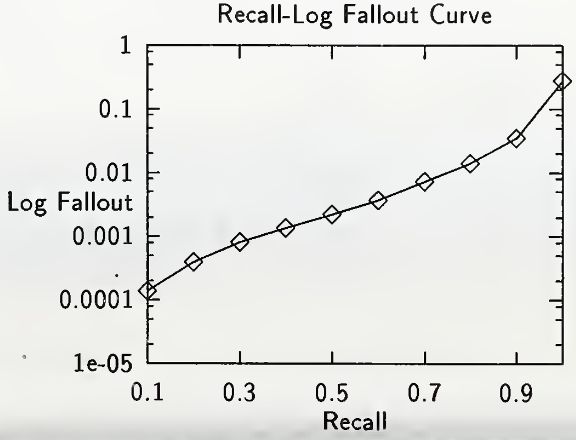
Document Level Averages	
	Precision
At 5 docs	0.5224
At 10 docs	0.4959
At 15 docs	0.4721
At 20 docs	0.4337
At 30 docs	0.3939
At 100 docs	0.2743
At 200 docs	0.2021
At 500 docs	0.1214
At 1000 docs	0.0777

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2977

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00014
0.20	0.00040
0.30	0.00082
0.40	0.00138
0.50	0.00223
0.60	0.00371
0.70	0.00735
0.80	0.01397
0.90	0.03527
1.00	0.27687







Summary Statistics	
Run Number	citym1–category A, manual
Number of Topics	49

Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	3539

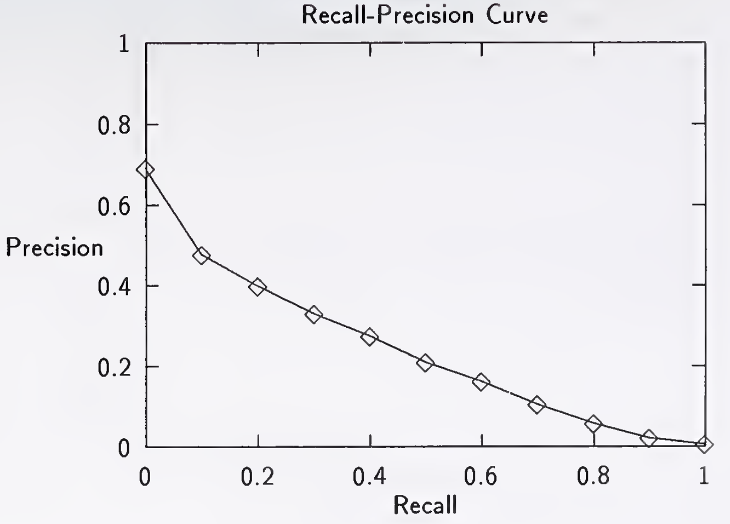
Recall Level Precision Averages	
Recall	Precision
0.00	0.6900
0.10	0.4775
0.20	0.3995
0.30	0.3302
0.40	0.2758
0.50	0.2105
0.60	0.1615
0.70	0.1051
0.80	0.0570
0.90	0.0203
1.00	0.0042

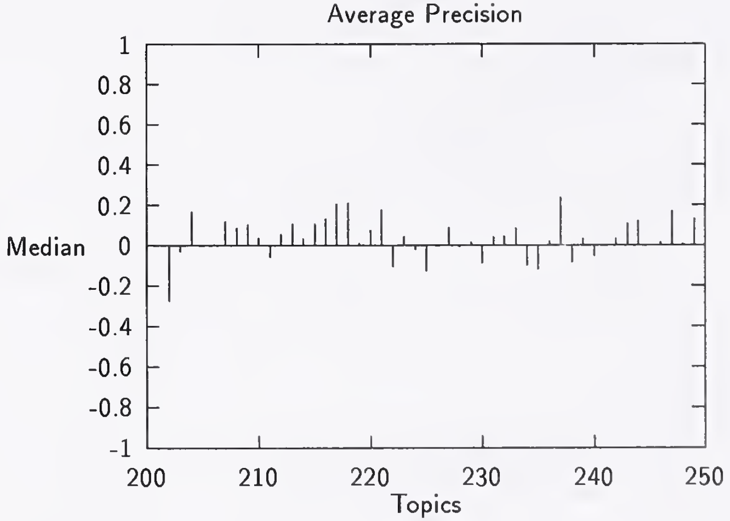
Average precision over all relevant docs	
non-interpolated	0.2257

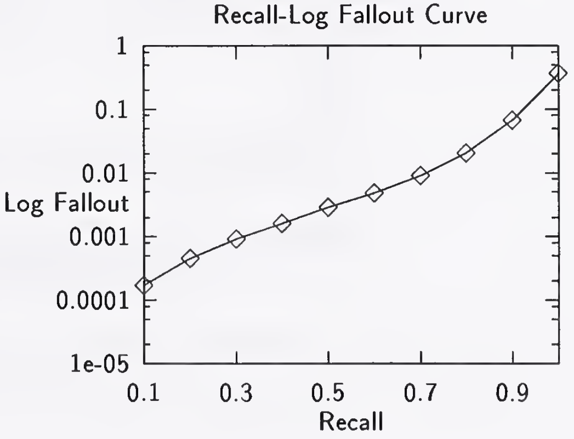
Document Level Averages	
	Precision
At 5 docs	0.4980
At 10 docs	0.4612
At 15 docs	0.4313
At 20 docs	0.4051
At 30 docs	0.3680
At 100 docs	0.2535
At 200 docs	0.1848
At 500 docs	0.1100
At 1000 docs	0.0722

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2713

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00017
0.20	0.00046
0.30	0.00093
0.40	0.00161
0.50	0.00287
0.60	0.00477
0.70	0.00913
0.80	0.02027
0.90	0.06651
1.00	0.36305





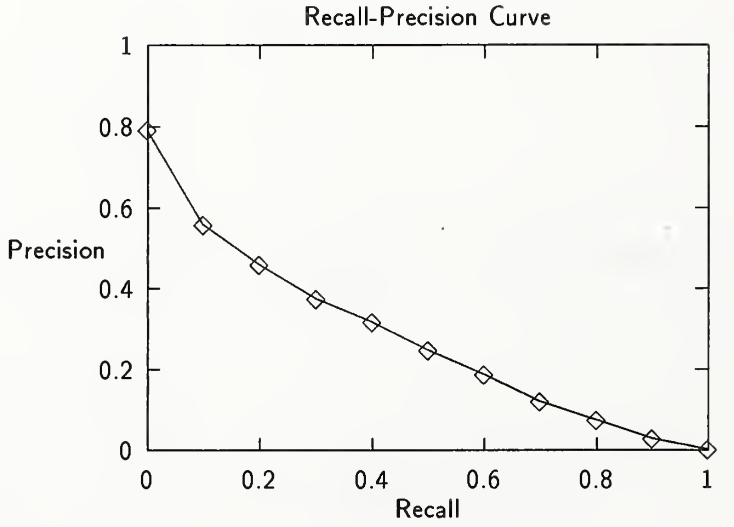


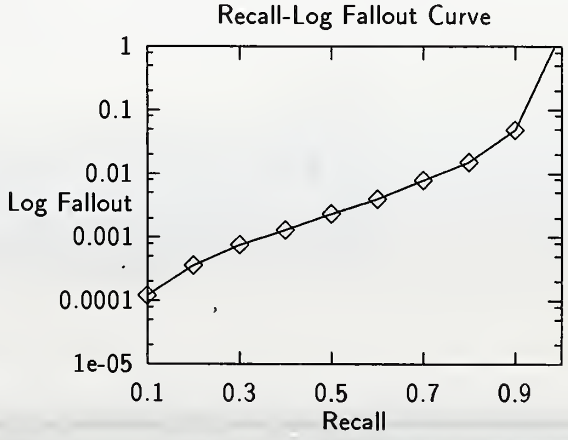
Summary Statistics	
Run Number	CLARTF–category A, manual
Number of Topics	49
Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	3973

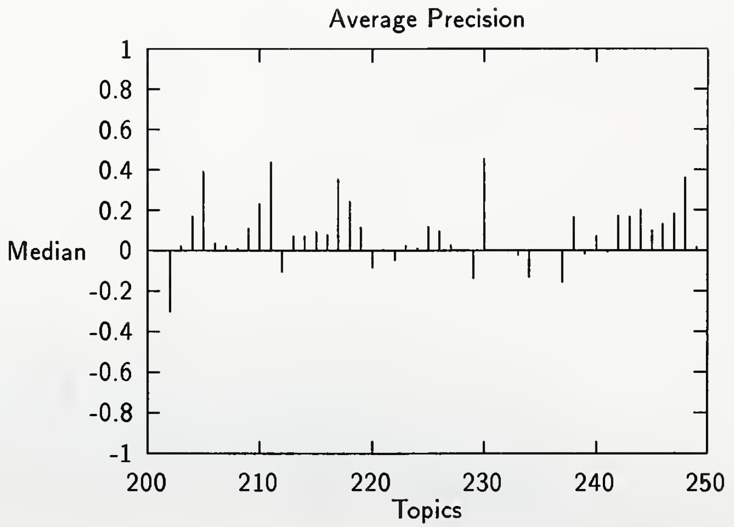
Recall Level Precision Averages	
Recall	Precision
0.00	0.7921
0.10	0.5580
0.20	0.4591
0.30	0.3747
0.40	0.3181
0.50	0.2485
0.60	0.1883
0.70	0.1212
0.80	0.0748
0.90	0.0279
1.00	0.0009
Average precision over all relevant docs	
non-interpolated	0.2669

Document Level Averages	
	Precision
At 5 docs	0.5469
At 10 docs	0.5184
At 15 docs	0.4912
At 20 docs	0.4673
At 30 docs	0.4347
At 100 docs	0.3063
At 200 docs	0.2200
At 500 docs	0.1313
At 1000 docs	0.0811
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3108

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00012
0.20	0.00036
0.30	0.00077
0.40	0.00131
0.50	0.00232
0.60	0.00396
0.70	0.00777
0.80	0.01515
0.90	0.04802
1.00	1.69984





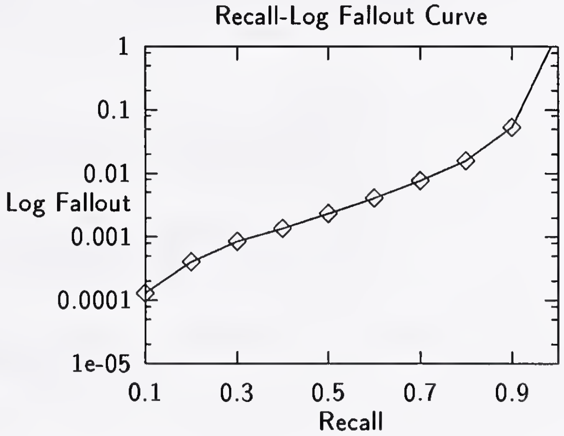


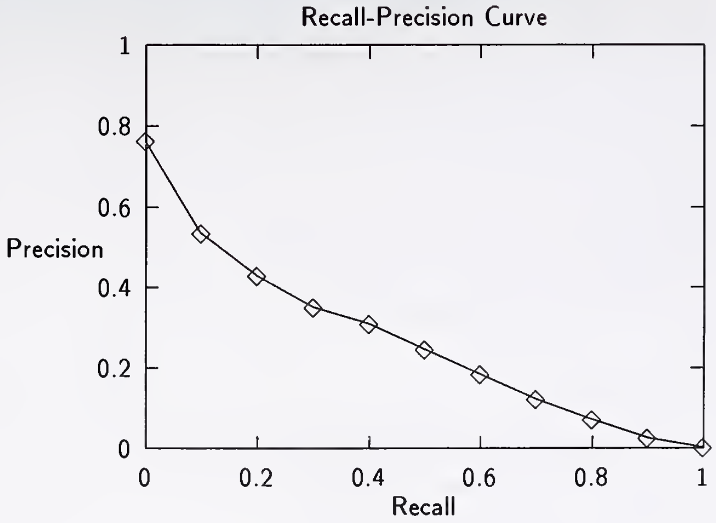
Summary Statistics	
Run Number	CLARTN–category A, manual
Number of Topics	49
Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	3869

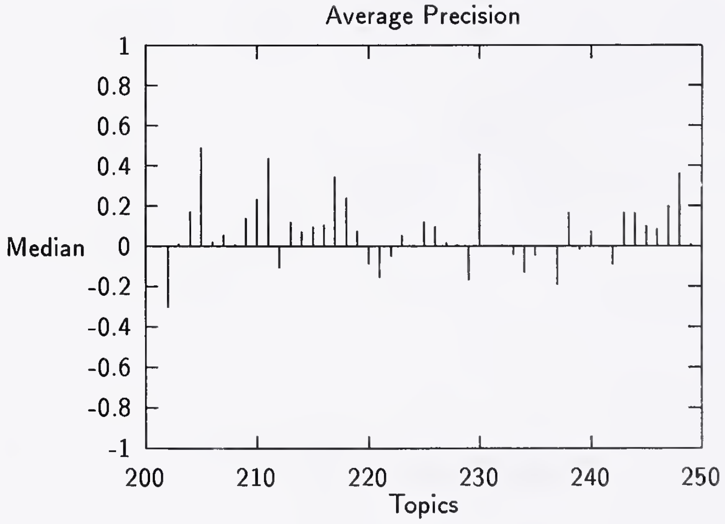
Recall Level Precision Averages	
Recall	Precision
0.00	0.7626
0.10	0.5340
0.20	0.4293
0.30	0.3517
0.40	0.3097
0.50	0.2460
0.60	0.1850
0.70	0.1224
0.80	0.0717
0.90	0.0256
1.00	0.0009
Average precision over all relevant docs	
non-interpolated	0.2576

Document Level Averages	
	Precision
At 5 docs	0.5388
At 10 docs	0.4837
At 15 docs	0.4707
At 20 docs	0.4500
At 30 docs	0.4143
At 100 docs	0.2953
At 200 docs	0.2134
At 500 docs	0.1279
At 1000 docs	0.0790
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3004

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00013
0.20	0.00041
0.30	0.00085
0.40	0.00137
0.50	0.00235
0.60	0.00405
0.70	0.00769
0.80	0.01586
0.90	0.05245
1.00	1.69984







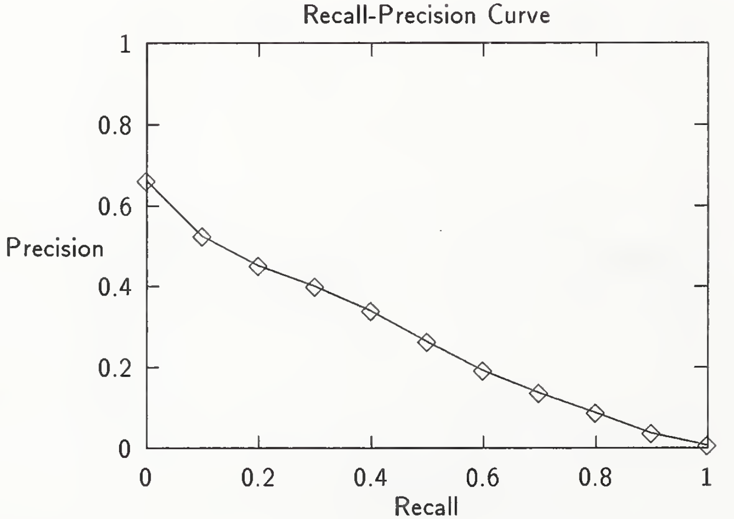
Summary Statistics	
Run Number	CnQst1–category A, manual
Number of Topics	49

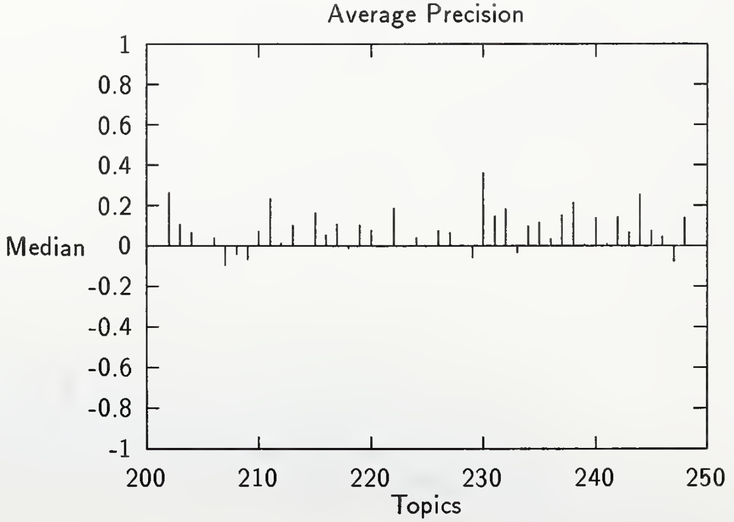
Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	4187

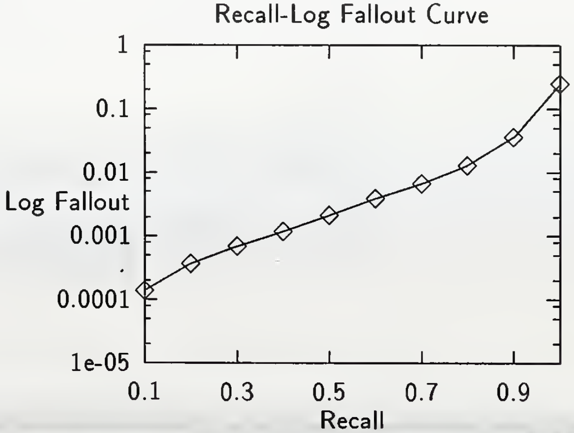
Recall Level Precision Averages	
Recall	Precision
0.00	0.6613
0.10	0.5248
0.20	0.4518
0.30	0.4004
0.40	0.3402
0.50	0.2645
0.60	0.1924
0.70	0.1374
0.80	0.0872
0.90	0.0371
1.00	0.0061
Average precision over all relevant docs	
non-interpolated	0.2626

Document Level Averages	
	Precision
At 5 docs	0.4327
At 10 docs	0.4531
At 15 docs	0.4517
At 20 docs	0.4449
At 30 docs	0.4204
At 100 docs	0.3071
At 200 docs	0.2303
At 500 docs	0.1382
At 1000 docs	0.0854
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3271

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00014
0.20	0.00037
0.30	0.00069
0.40	0.00119
0.50	0.00213
0.60	0.00386
0.70	0.00673
0.80	0.01282
0.90	0.03577
1.00	0.24949







Summary Statistics	
Run Number	CnQst2–category A, manual
Number of Topics	49

Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	4614

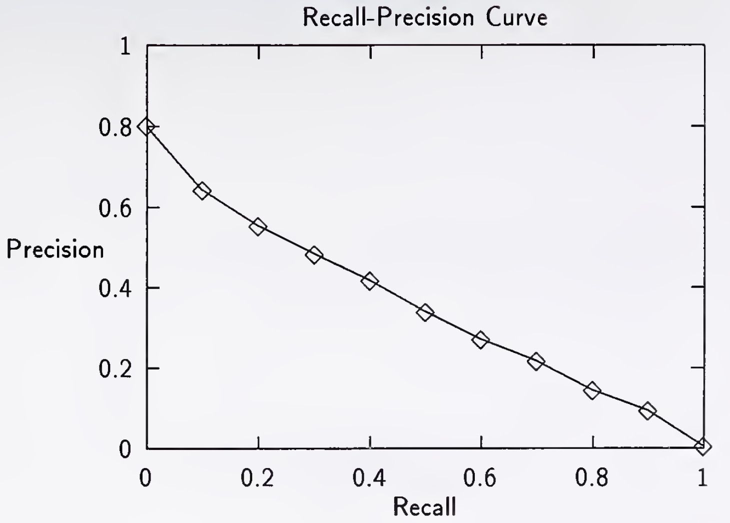
Recall Level Precision Averages	
Recall	Precision
0.00	0.8020
0.10	0.6415
0.20	0.5535
0.30	0.4839
0.40	0.4178
0.50	0.3407
0.60	0.2726
0.70	0.2183
0.80	0.1440
0.90	0.0932
1.00	0.0043

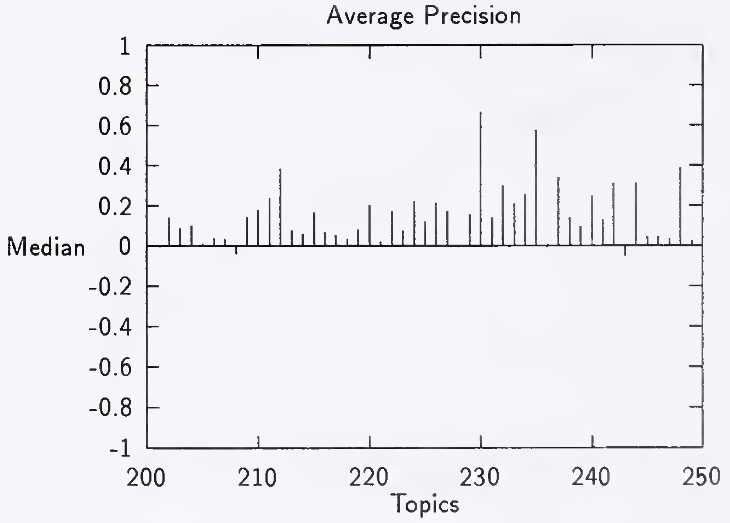
Average precision over all relevant docs	
non-interpolated	0.3436

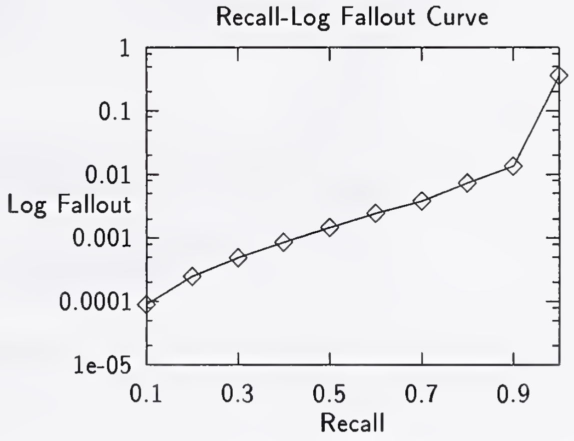
Document Level Averages	
	Precision
At 5 docs	0.5714
At 10 docs	0.5653
At 15 docs	0.5197
At 20 docs	0.5061
At 30 docs	0.4830
At 100 docs	0.3716
At 200 docs	0.2694
At 500 docs	0.1567
At 1000 docs	0.0942

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3785

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00009
0.20	0.00025
0.30	0.00049
0.40	0.00085
0.50	0.00148
0.60	0.00245
0.70	0.00384
0.80	0.00728
0.90	0.01341
1.00	0.35457





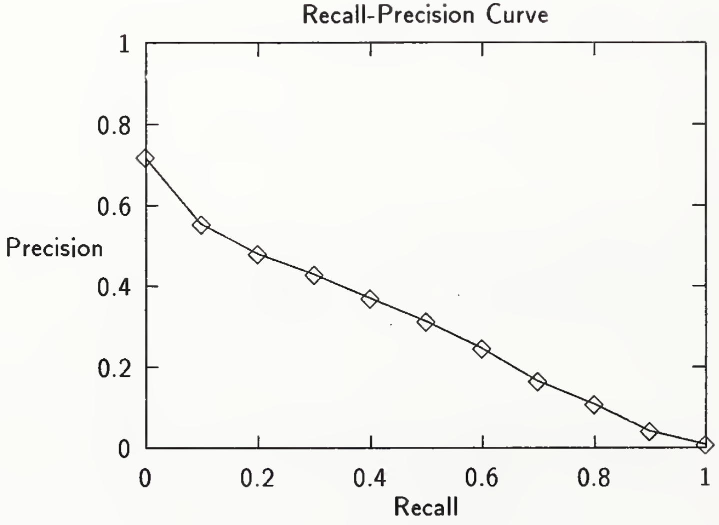


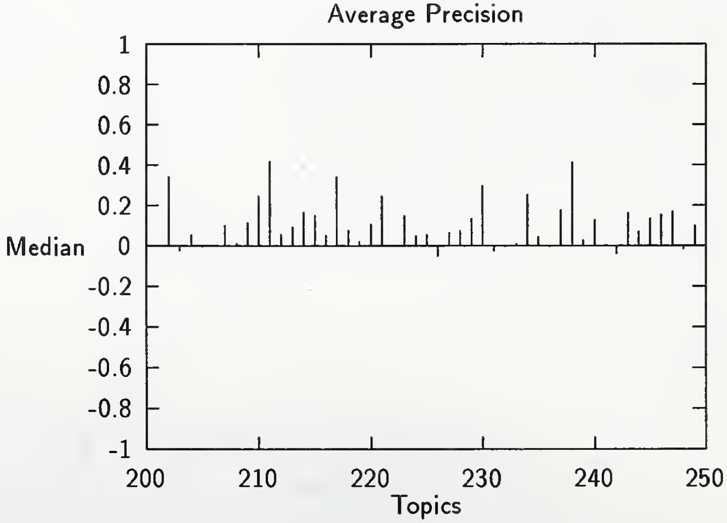
Summary Statistics	
Run Number	CrnlAE–category A, automatic
Number of Topics	49
Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	4350

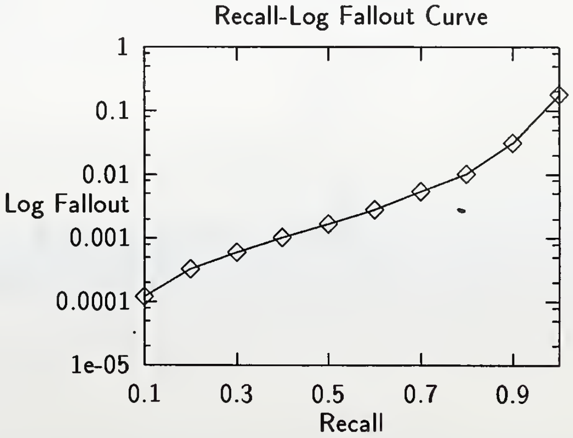
Recall Level Precision Averages	
Recall	Precision
0.00	0.7184
0.10	0.5527
0.20	0.4807
0.30	0.4300
0.40	0.3708
0.50	0.3129
0.60	0.2461
0.70	0.1653
0.80	0.1078
0.90	0.0421
1.00	0.0086
Average precision over all relevant docs	
non-interpolated	0.2944

Document Level Averages	
	Precision
At 5 docs	0.5265
At 10 docs	0.5102
At 15 docs	0.4939
At 20 docs	0.4684
At 30 docs	0.4361
At 100 docs	0.3112
At 200 docs	0.2364
At 500 docs	0.1433
At 1000 docs	0.0888
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3384

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00012
0.20	0.00033
0.30	0.00061
0.40	0.00104
0.50	0.00168
0.60	0.00281
0.70	0.00541
0.80	0.01014
0.90	0.03136
1.00	0.17652





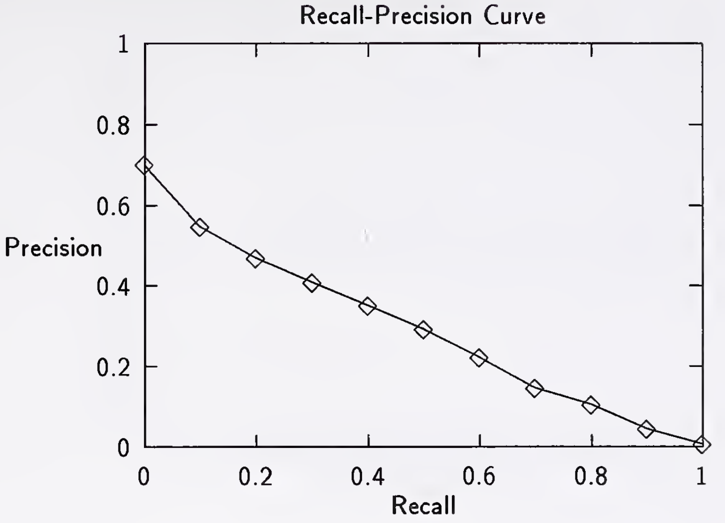


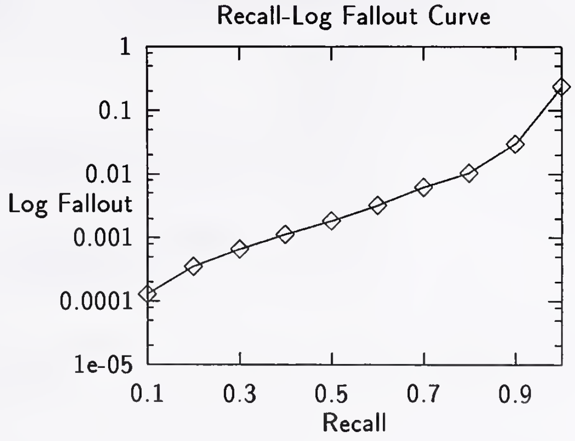
Summary Statistics	
Run Number	CrnlAL–category A, automatic
Number of Topics	49
Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	4297

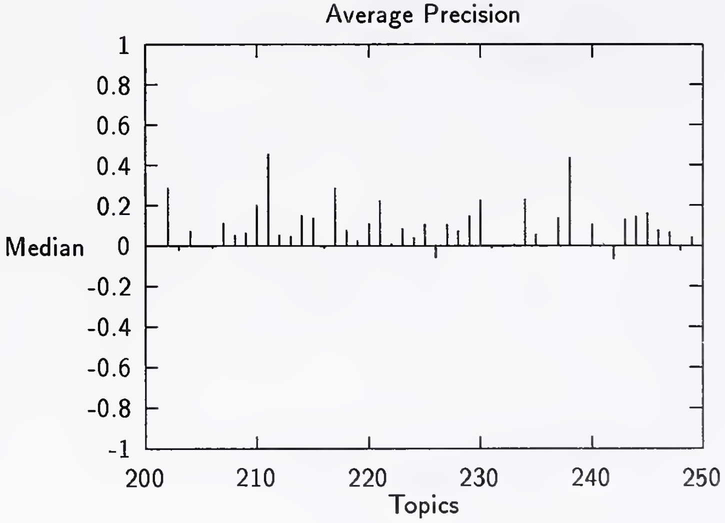
Recall Level Precision Averages	
Recall	Precision
0.00	0.7018
0.10	0.5463
0.20	0.4687
0.30	0.4086
0.40	0.3519
0.50	0.2927
0.60	0.2227
0.70	0.1461
0.80	0.1053
0.90	0.0449
1.00	0.0065
Average precision over all relevant docs	
non-interpolated	0.2829

Document Level Averages	
	Precision
At 5 docs	0.5102
At 10 docs	0.4980
At 15 docs	0.4884
At 20 docs	0.4561
At 30 docs	0.4197
At 100 docs	0.3002
At 200 docs	0.2334
At 500 docs	0.1420
At 1000 docs	0.0877
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3256

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00013
0.20	0.00035
0.30	0.00066
0.40	0.00113
0.50	0.00185
0.60	0.00321
0.70	0.00626
0.80	0.01041
0.90	0.02931
1.00	0.23404







Summary Statistics	
Run Number	DCU951–category A, manual
Number of Topics	49

Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	2482

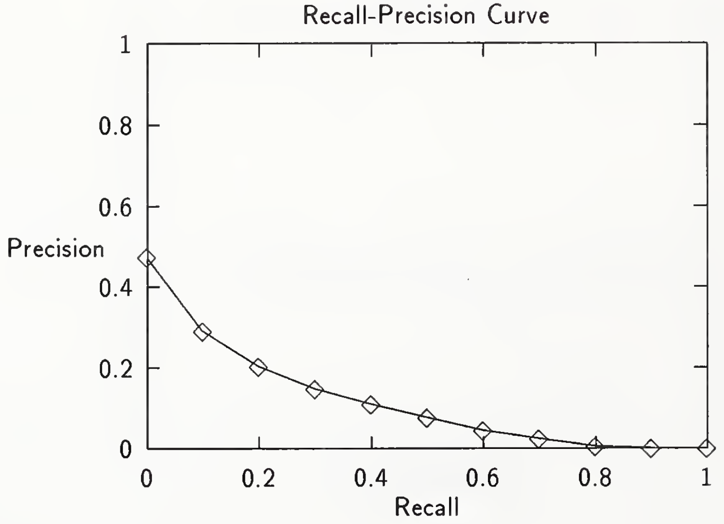
Recall Level Precision Averages	
Recall	Precision
0.00	0.4740
0.10	0.2907
0.20	0.2040
0.30	0.1471
0.40	0.1093
0.50	0.0760
0.60	0.0453
0.70	0.0246
0.80	0.0050
0.90	0.0000
1.00	0.0000

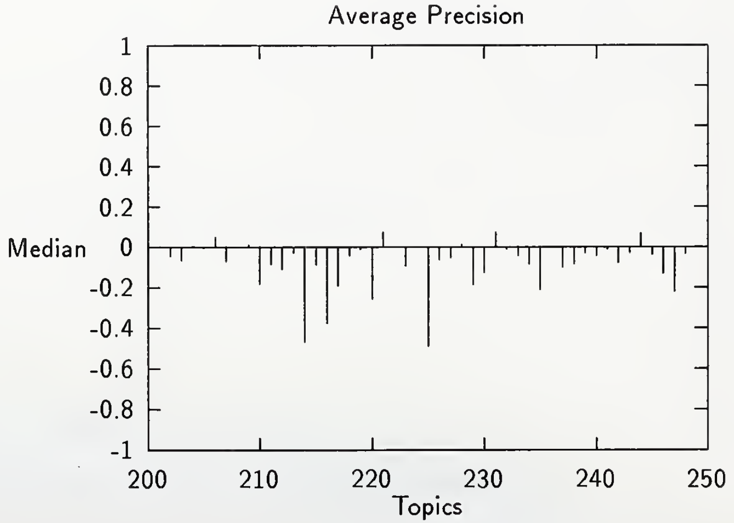
Average precision over all relevant docs	
non-interpolated	0.1066

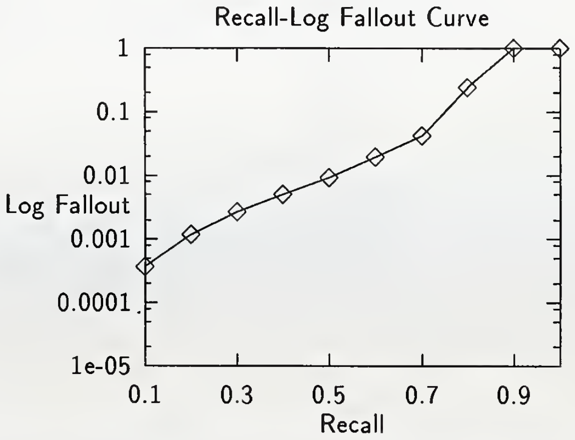
Document Level Averages	
	Precision
At 5 docs	0.2857
At 10 docs	0.2755
At 15 docs	0.2517
At 20 docs	0.2418
At 30 docs	0.2136
At 100 docs	0.1537
At 200 docs	0.1191
At 500 docs	0.0757
At 1000 docs	0.0507

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1623

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00037
0.20	0.00119
0.30	0.00266
0.40	0.00499
0.50	0.00931
0.60	0.01936
0.70	0.04250
0.80	0.24377
0.90	1.00000
1.00	1.00000







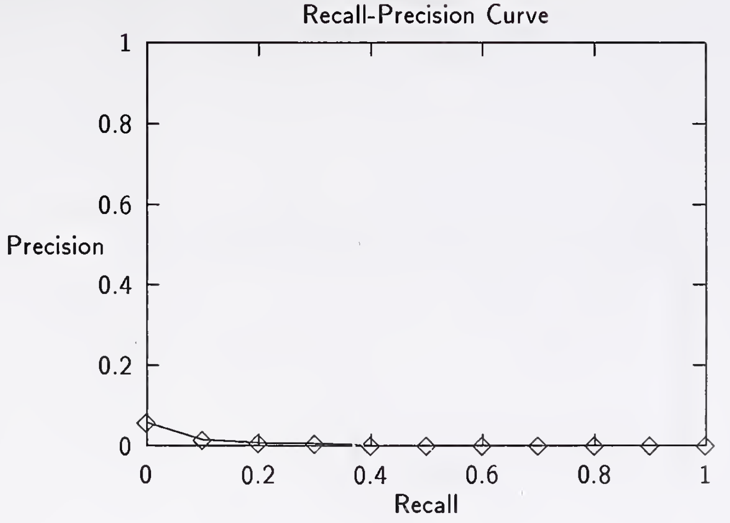
Summary Statistics	
Run Number	DCU952–category A, manual
Number of Topics	49

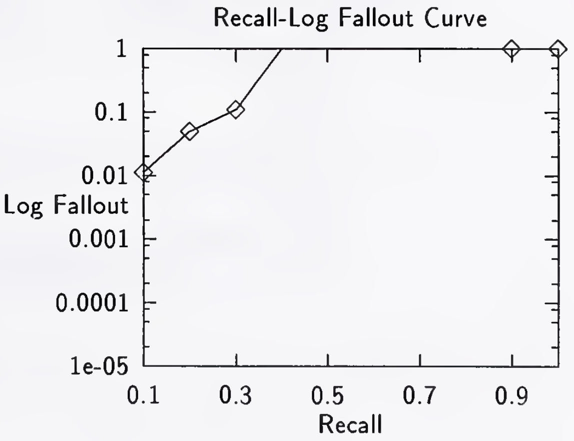
Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	347

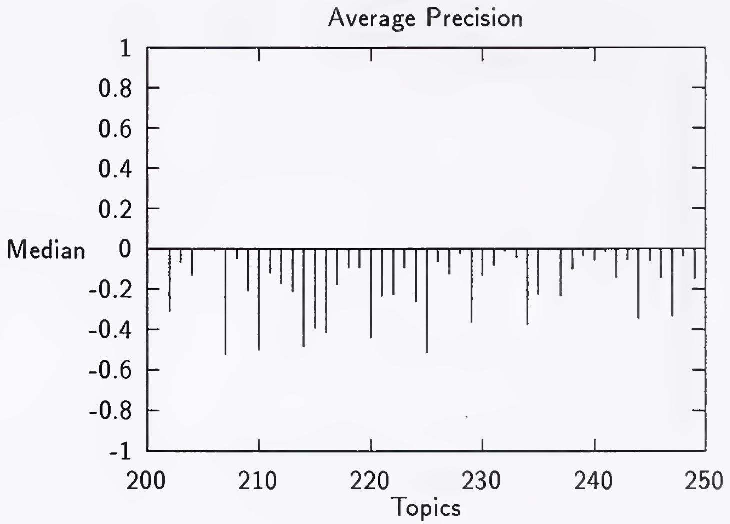
Recall Level Precision Averages	
Recall	Precision
0.00	0.0577
0.10	0.0136
0.20	0.0061
0.30	0.0042
0.40	0.0006
0.50	0.0001
0.60	0.0001
0.70	0.0001
0.80	0.0001
0.90	0.0000
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.0037

Document Level Averages	
	Precision
At 5 docs	0.0122
At 10 docs	0.0184
At 15 docs	0.0218
At 20 docs	0.0214
At 30 docs	0.0197
At 100 docs	0.0153
At 200 docs	0.0115
At 500 docs	0.0087
At 1000 docs	0.0071
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0129

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.01111
0.20	0.04990
0.30	0.10891
0.40	1.02021
0.50	7.65540
0.60	9.18648
0.70	10.71756
0.80	12.24864
0.90	1.00000
1.00	1.00000





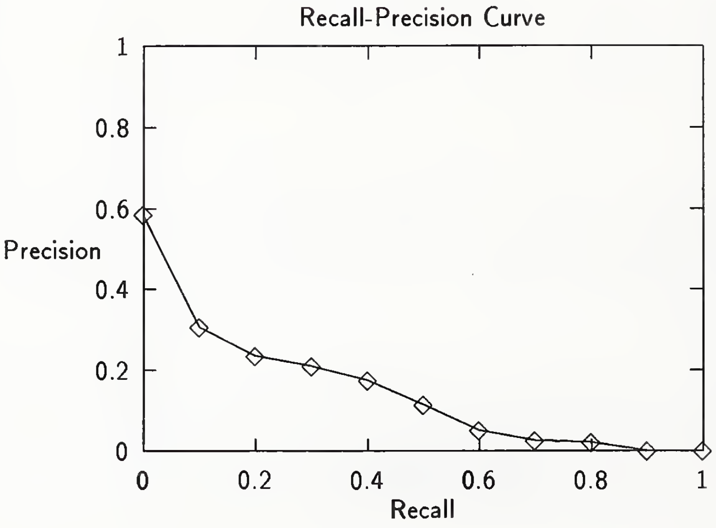


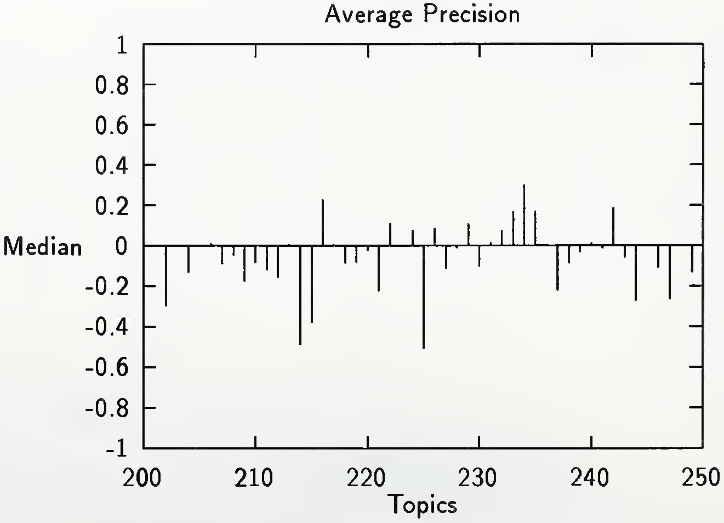
Summary Statistics	
Run Number	fsclt1–category A, manual
Number of Topics	49
Total number of documents over all topics	
Retrieved:	11472
Relevant:	6501
Rel_ret:	1103

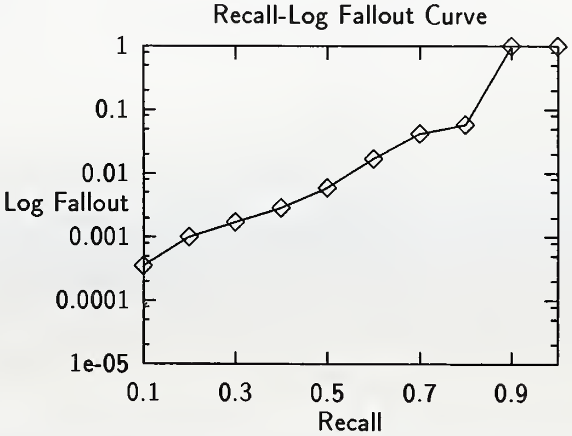
Recall Level Precision Averages	
Recall	Precision
0.00	0.5858
0.10	0.3066
0.20	0.2348
0.30	0.2096
0.40	0.1753
0.50	0.1146
0.60	0.0513
0.70	0.0250
0.80	0.0204
0.90	0.0000
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.1303

Document Level Averages	
	Precision
At 5 docs	0.3918
At 10 docs	0.3571
At 15 docs	0.3306
At 20 docs	0.3122
At 30 docs	0.2810
At 100 docs	0.1524
At 200 docs	0.0913
At 500 docs	0.0425
At 1000 docs	0.0225
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1840

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00035
0.20	0.00100
0.30	0.00173
0.40	0.00288
0.50	0.00592
0.60	0.01699
0.70	0.04180
0.80	0.05882
0.90	1.00000
1.00	1.00000







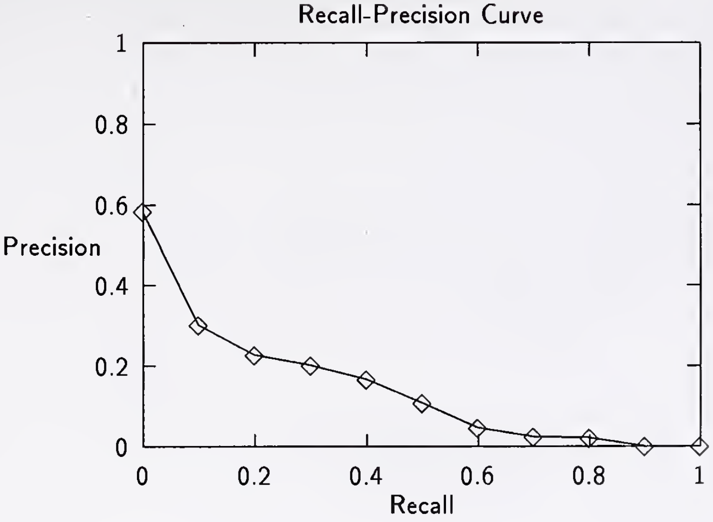
Summary Statistics	
Run Number	fsclt2–category A, manual
Number of Topics	49

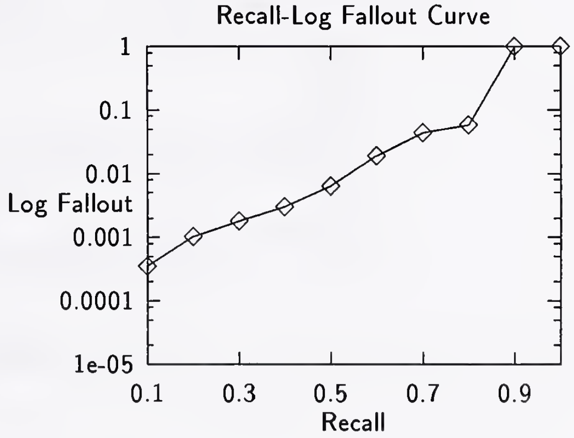
Total number of documents over all topics	
Retrieved:	11472
Relevant:	6501
Rel_ret:	1108

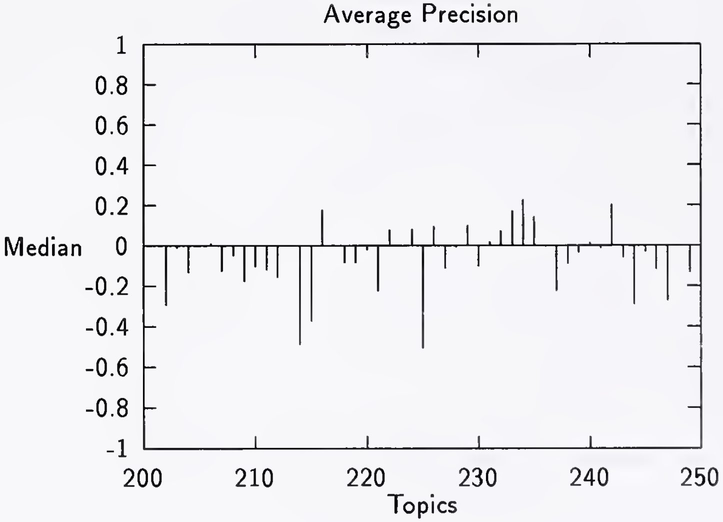
Recall Level Precision Averages	
Recall	Precision
0.00	0.5830
0.10	0.3028
0.20	0.2275
0.30	0.2012
0.40	0.1673
0.50	0.1077
0.60	0.0465
0.70	0.0238
0.80	0.0206
0.90	0.0000
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.1248

Document Level Averages	
	Precision
At 5 docs	0.3755
At 10 docs	0.3510
At 15 docs	0.3238
At 20 docs	0.3112
At 30 docs	0.2796
At 100 docs	0.1504
At 200 docs	0.0888
At 500 docs	0.0419
At 1000 docs	0.0226
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1826

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00035
0.20	0.00104
0.30	0.00182
0.40	0.00305
0.50	0.00634
0.60	0.01884
0.70	0.04396
0.80	0.05824
0.90	1.00000
1.00	1.00000







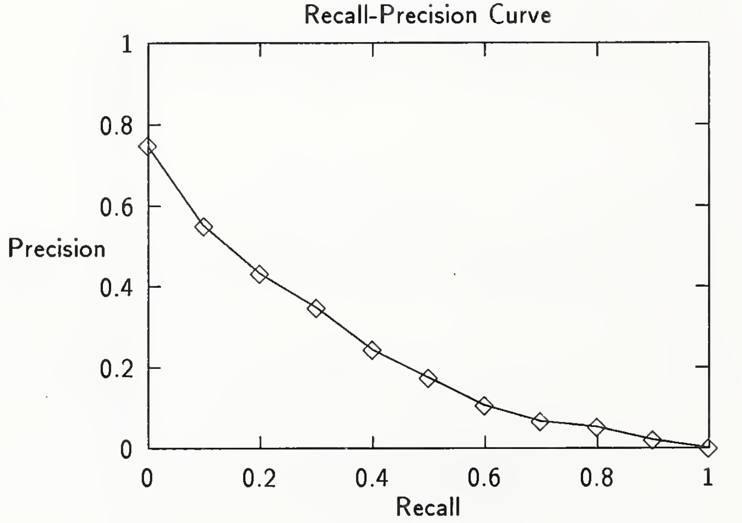
Summary Statistics	
Run Number	gmu1–category A, manual
Number of Topics	49

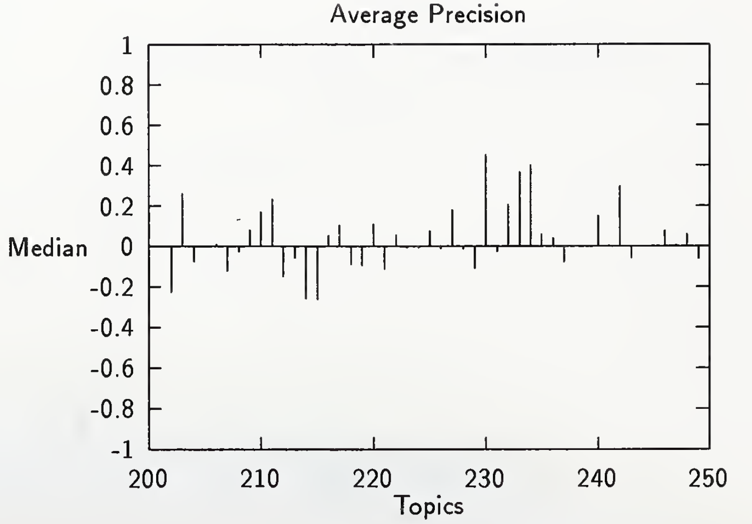
Total number of documents over all topics	
Retrieved:	15402
Relevant:	6501
Rel_ret:	2386

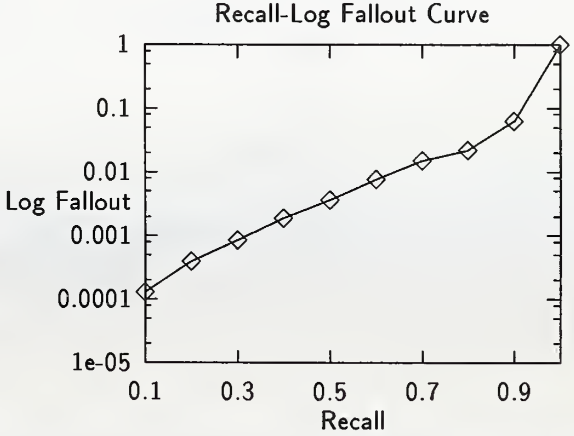
Recall Level Precision Averages	
Recall	Precision
0.00	0.7488
0.10	0.5501
0.20	0.4325
0.30	0.3483
0.40	0.2451
0.50	0.1748
0.60	0.1062
0.70	0.0667
0.80	0.0526
0.90	0.0214
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.2216

Document Level Averages	
	Precision
At 5 docs	0.5102
At 10 docs	0.4918
At 15 docs	0.4844
At 20 docs	0.4592
At 30 docs	0.4299
At 100 docs	0.2904
At 200 docs	0.1869
At 500 docs	0.0920
At 1000 docs	0.0487
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2931

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00013
0.20	0.00040
0.30	0.00086
0.40	0.00189
0.50	0.00361
0.60	0.00773
0.70	0.01500
0.80	0.02206
0.90	0.06302
1.00	1.00000





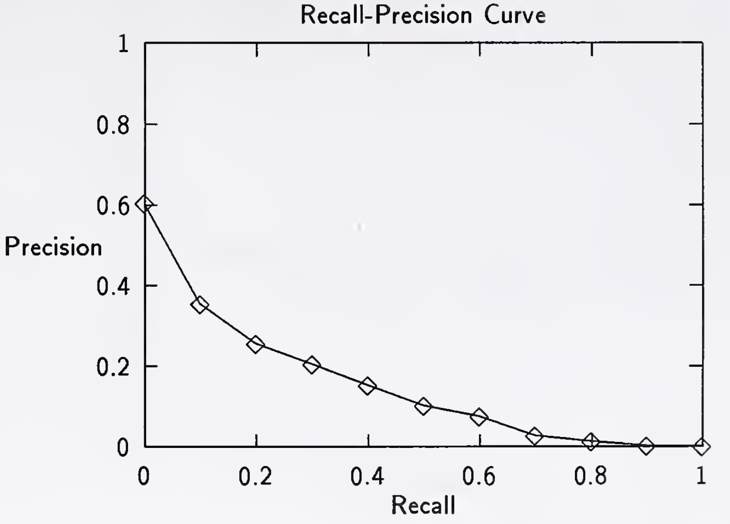


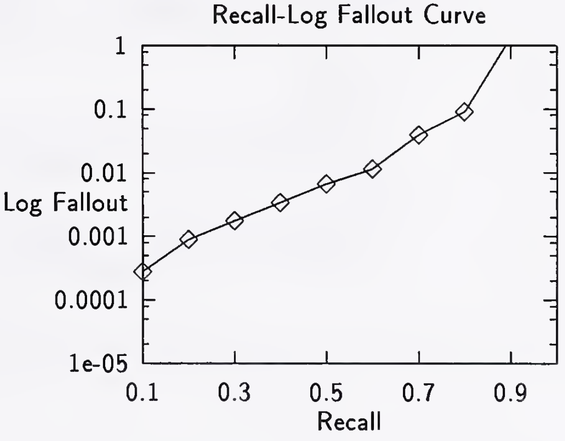
Summary Statistics	
Run Number	gmu2–category A, automatic
Number of Topics	49
Total number of documents over all topics	
Retrieved:	48999
Relevant:	6501
Rel_ret:	2685

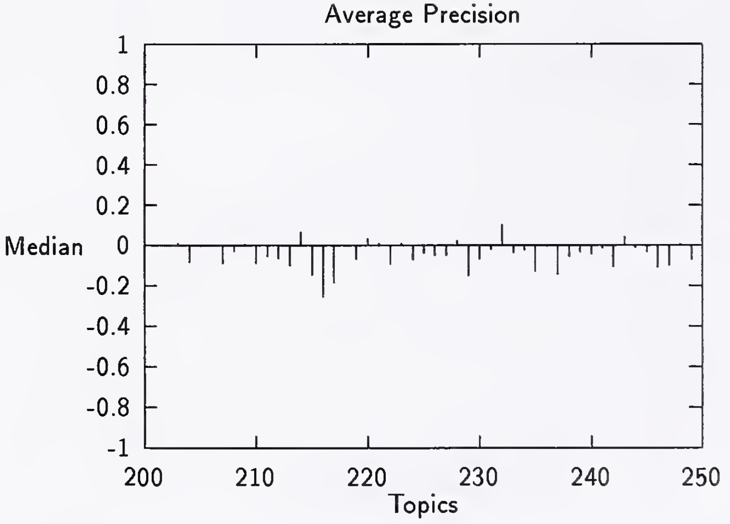
Recall Level Precision Averages	
Recall	Precision
0.00	0.6034
0.10	0.3552
0.20	0.2563
0.30	0.2056
0.40	0.1522
0.50	0.1024
0.60	0.0754
0.70	0.0268
0.80	0.0133
0.90	0.0010
1.00	0.0001
Average precision over all relevant docs	
non-interpolated	0.1385

Document Level Averages	
	Precision
At 5 docs	0.3551
At 10 docs	0.3449
At 15 docs	0.3129
At 20 docs	0.2806
At 30 docs	0.2565
At 100 docs	0.1802
At 200 docs	0.1377
At 500 docs	0.0857
At 1000 docs	0.0548
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2005

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00028
0.20	0.00089
0.30	0.00177
0.40	0.00341
0.50	0.00671
0.60	0.01127
0.70	0.03892
0.80	0.09088
0.90	1.37673
1.00	15.31080







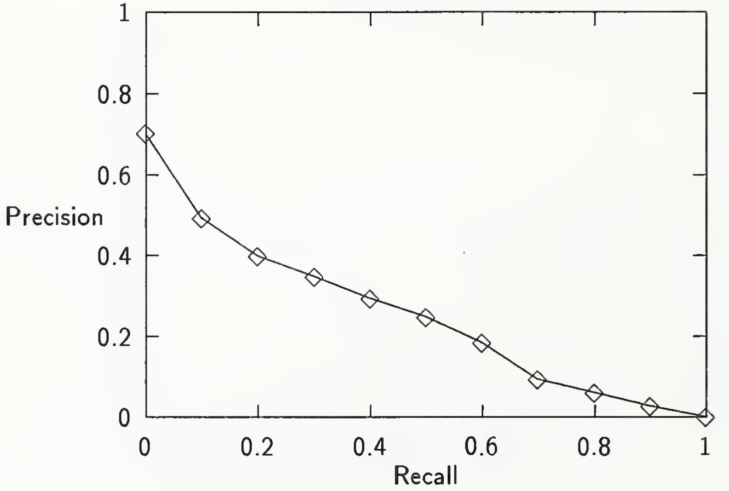
Summary Statistics	
Run Number	INQ201–category A, automatic
Number of Topics	49
Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	3547

Recall Level Precision Averages	
Recall	Precision
0.00	0.7029
0.10	0.4929
0.20	0.3993
0.30	0.3484
0.40	0.2936
0.50	0.2475
0.60	0.1846
0.70	0.0933
0.80	0.0606
0.90	0.0276
1.00	0.0007
Average precision over all relevant docs	
non-interpolated	0.2407

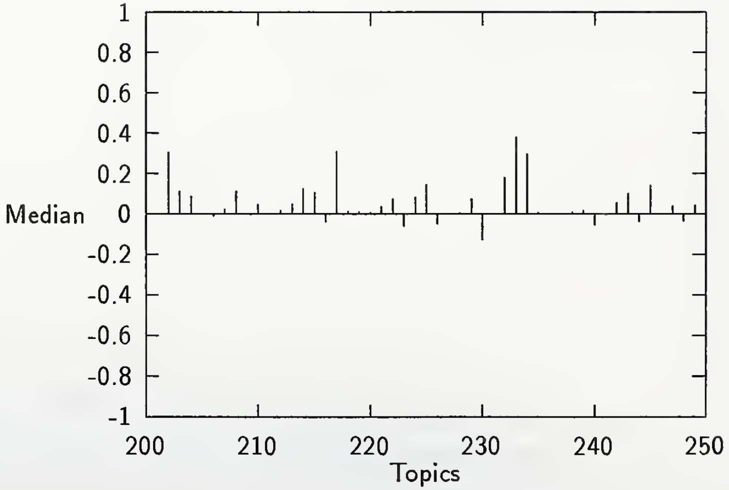
Document Level Averages	
	Precision
At 5 docs	0.5102
At 10 docs	0.4653
At 15 docs	0.4150
At 20 docs	0.3980
At 30 docs	0.3531
At 100 docs	0.2514
At 200 docs	0.1907
At 500 docs	0.1166
At 1000 docs	0.0724
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2904

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00016
0.20	0.00046
0.30	0.00086
0.40	0.00147
0.50	0.00233
0.60	0.00406
0.70	0.01042
0.80	0.01899
0.90	0.04855
1.00	2.18594

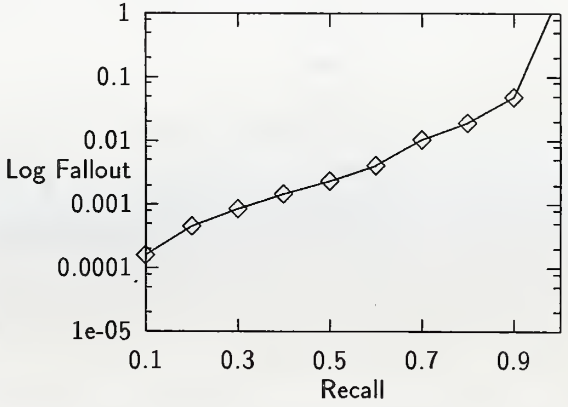
Recall-Precision Curve



Average Precision



Recall-Log Fallout Curve



Summary Statistics	
Run Number	INQ202–category A, manual
Number of Topics	49

Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	4067

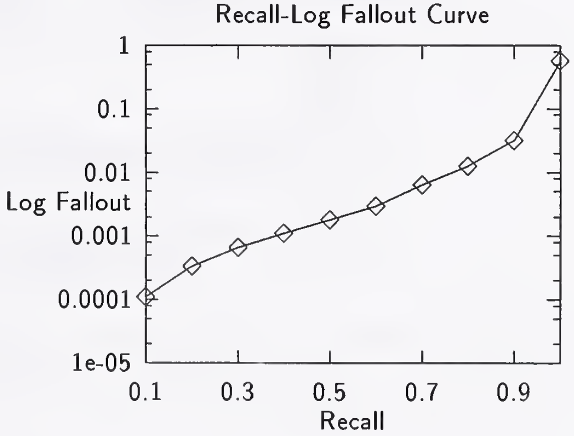
Recall Level Precision Averages	
Recall	Precision
0.00	0.7973
0.10	0.5878
0.20	0.4726
0.30	0.4079
0.40	0.3541
0.50	0.2982
0.60	0.2357
0.70	0.1454
0.80	0.0884
0.90	0.0415
1.00	0.0027

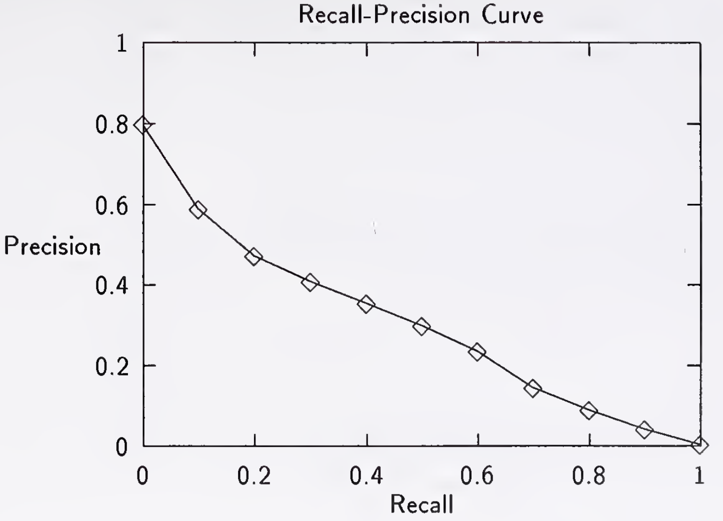
Average precision over all relevant docs	
non-interpolated	0.2913

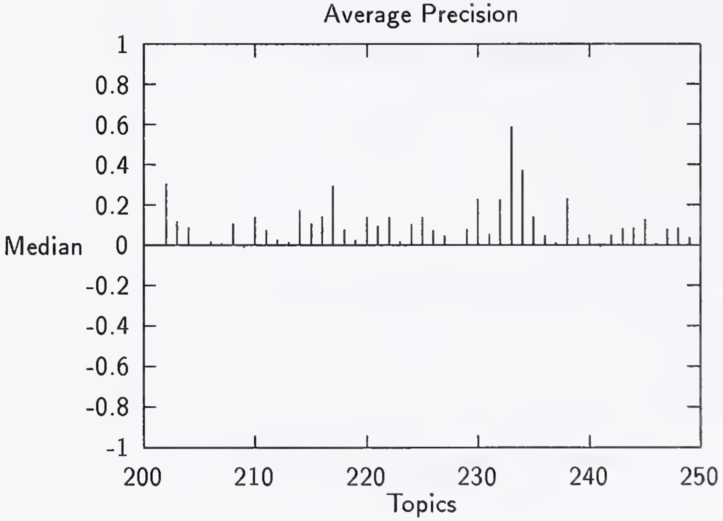
Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00011
0.20	0.00034
0.30	0.00067
0.40	0.00112
0.50	0.00180
0.60	0.00298
0.70	0.00630
0.80	0.01263
0.90	0.03183
1.00	0.56559

Document Level Averages	
	Precision
At 5 docs	0.5959
At 10 docs	0.5510
At 15 docs	0.5061
At 20 docs	0.4745
At 30 docs	0.4429
At 100 docs	0.3069
At 200 docs	0.2305
At 500 docs	0.1349
At 1000 docs	0.0830

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3425







Summary Statistics	
Run Number	INTXT2–category A, manual
Number of Topics	49

Total number of documents over all topics	
Retrieved:	8992
Relevant:	6501
Rel_ret:	2161

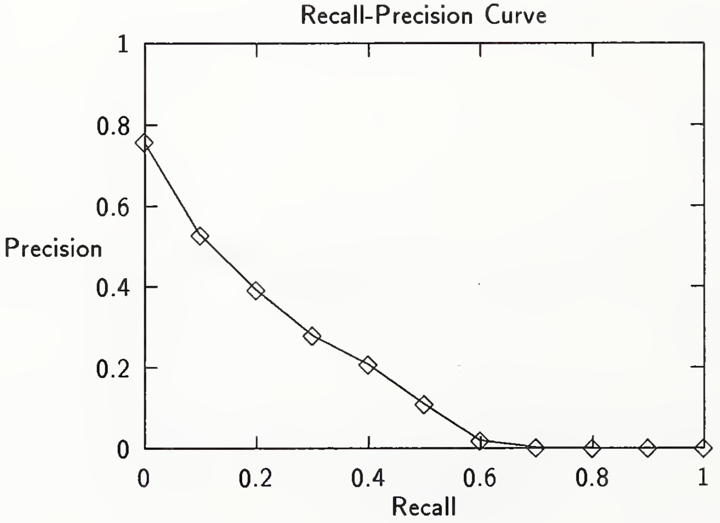
Recall Level Precision Averages	
Recall	Precision
0.00	0.7582
0.10	0.5283
0.20	0.3931
0.30	0.2791
0.40	0.2084
0.50	0.1091
0.60	0.0196
0.70	0.0014
0.80	0.0000
0.90	0.0000
1.00	0.0000

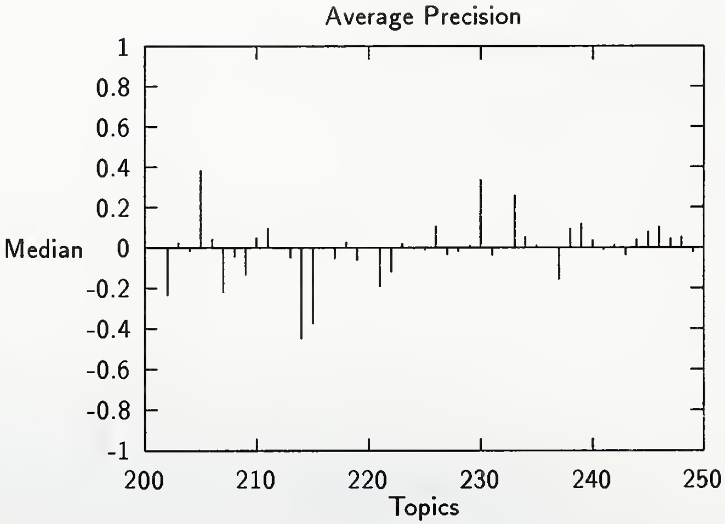
Average precision over all relevant docs	
non-interpolated	0.1814

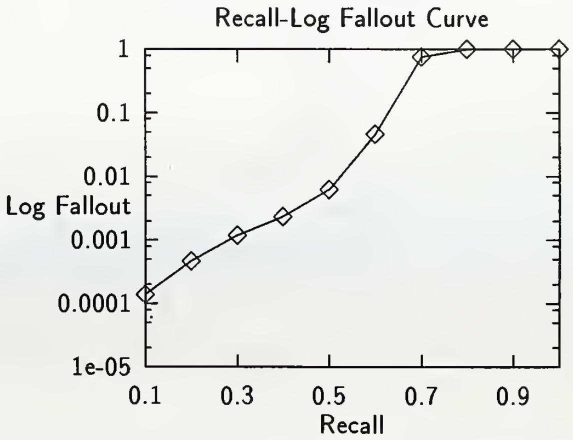
Document Level Averages	
	Precision
At 5 docs	0.5469
At 10 docs	0.5102
At 15 docs	0.4776
At 20 docs	0.4480
At 30 docs	0.4211
At 100 docs	0.2778
At 200 docs	0.1821
At 500 docs	0.0867
At 1000 docs	0.0441

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2615

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00014
0.20	0.00047
0.30	0.00119
0.40	0.00233
0.50	0.00625
0.60	0.04596
0.70	0.76454
0.80	1.00000
0.90	1.00000
1.00	1.00000





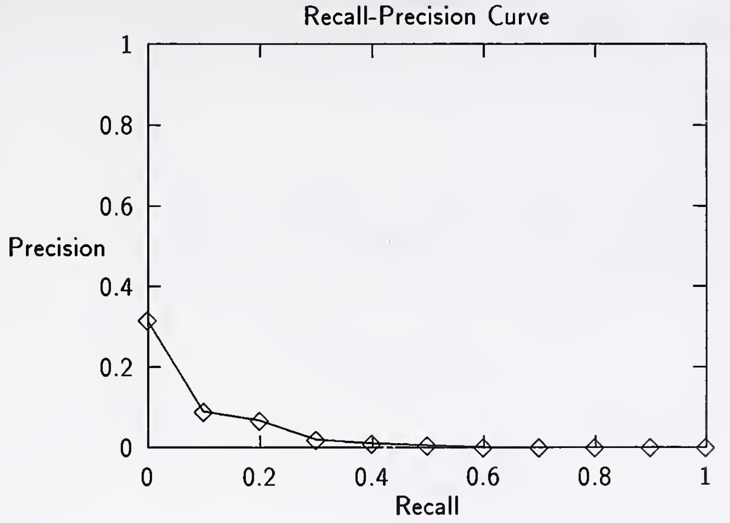


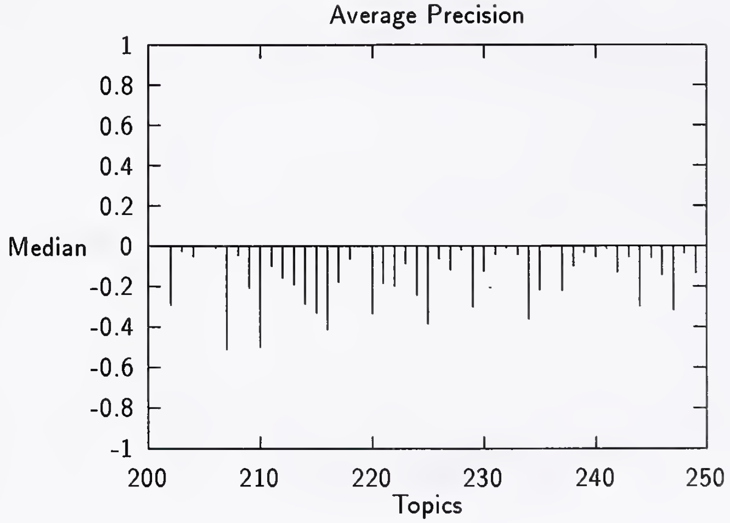
Summary Statistics	
Run Number	issah1–category A, automatic
Number of Topics	49
Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	1021

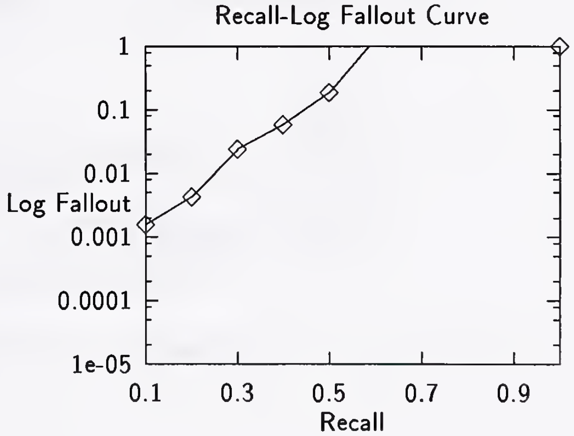
Recall Level Precision Averages	
Recall	Precision
0.00	0.3170
0.10	0.0893
0.20	0.0668
0.30	0.0186
0.40	0.0103
0.50	0.0041
0.60	0.0007
0.70	0.0007
0.80	0.0007
0.90	0.0007
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.0273

Document Level Averages	
	Precision
At 5 docs	0.1429
At 10 docs	0.1204
At 15 docs	0.1034
At 20 docs	0.1031
At 30 docs	0.0946
At 100 docs	0.0653
At 200 docs	0.0502
At 500 docs	0.0321
At 1000 docs	0.0208
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0697

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00156
0.20	0.00428
0.30	0.02424
0.40	0.05885
0.50	0.18597
0.60	1.31157
0.70	1.53016
0.80	1.74876
0.90	1.96735
1.00	1.00000







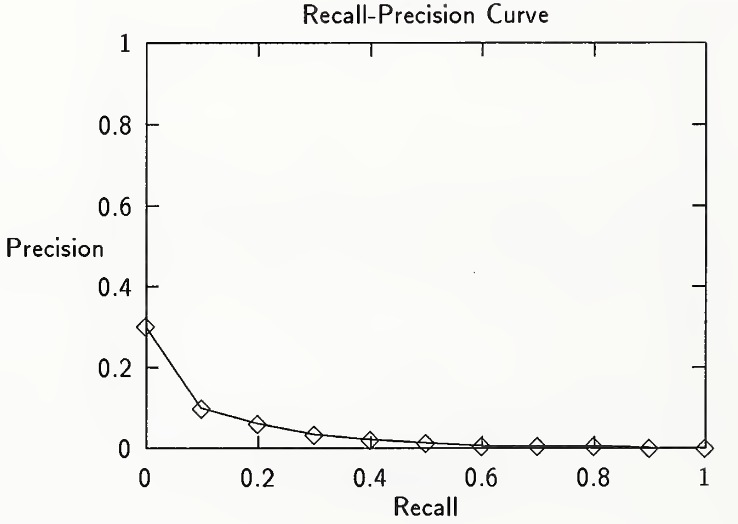
Summary Statistics	
Run Number	issah2–category A, manual
Number of Topics	49

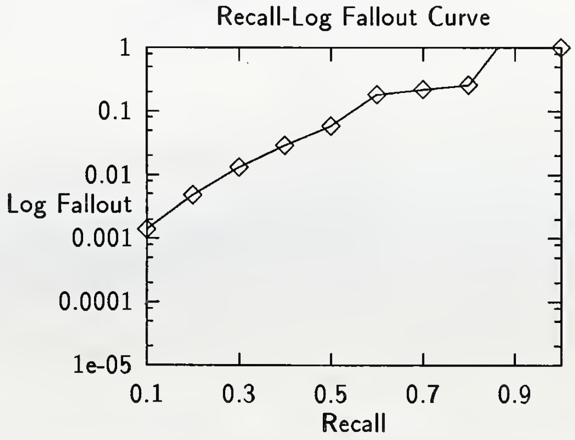
Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	987

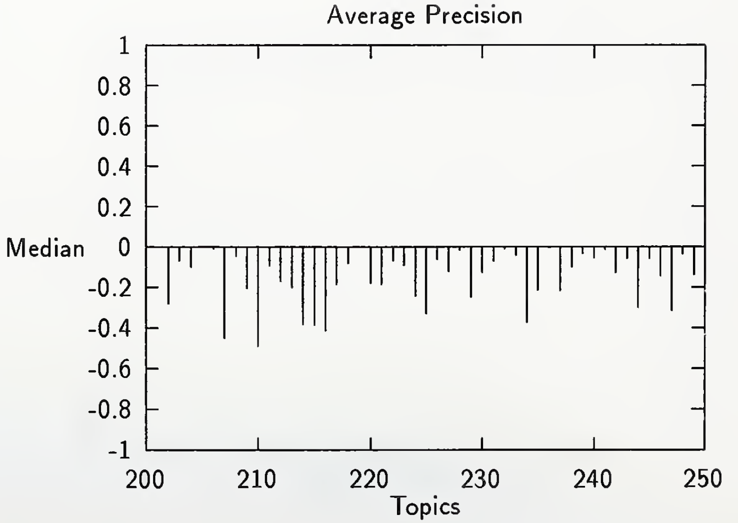
Recall Level Precision Averages	
Recall	Precision
0.00	0.3025
0.10	0.0978
0.20	0.0602
0.30	0.0339
0.40	0.0207
0.50	0.0131
0.60	0.0050
0.70	0.0049
0.80	0.0047
0.90	0.0006
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.0307

Document Level Averages	
	Precision
At 5 docs	0.1551
At 10 docs	0.1224
At 15 docs	0.1075
At 20 docs	0.0959
At 30 docs	0.0884
At 100 docs	0.0686
At 200 docs	0.0543
At 500 docs	0.0337
At 1000 docs	0.0201
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0670

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00141
0.20	0.00478
0.30	0.01309
0.40	0.02898
0.50	0.05768
0.60	0.18283
0.70	0.21768
0.80	0.25941
0.90	2.29547
1.00	1.00000







Summary Statistics	
Run Number	nyuge3–category A, automatic
Number of Topics	49

Total number of documents over all topics	
Retrieved:	46997
Relevant:	6501
Rel_ret:	2493

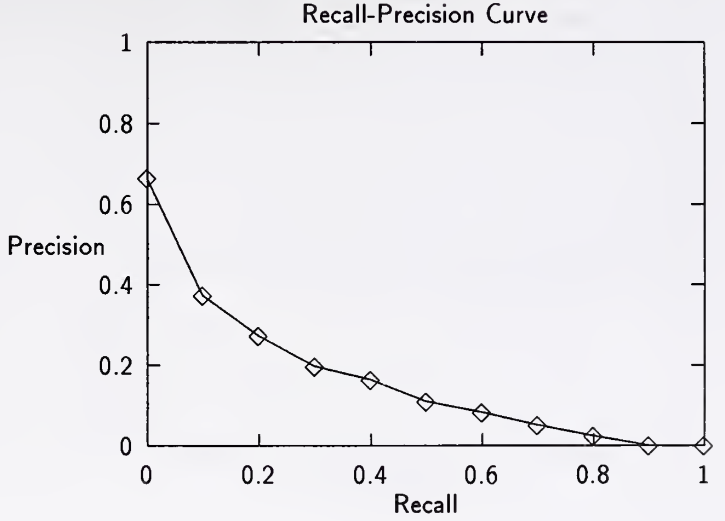
Recall Level Precision Averages	
Recall	Precision
0.00	0.6646
0.10	0.3733
0.20	0.2737
0.30	0.1971
0.40	0.1641
0.50	0.1094
0.60	0.0824
0.70	0.0505
0.80	0.0233
0.90	0.0007
1.00	0.0000

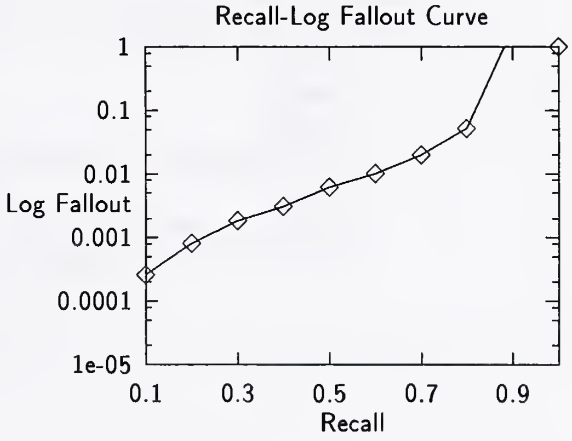
Average precision over all relevant docs	
non-interpolated	0.1501

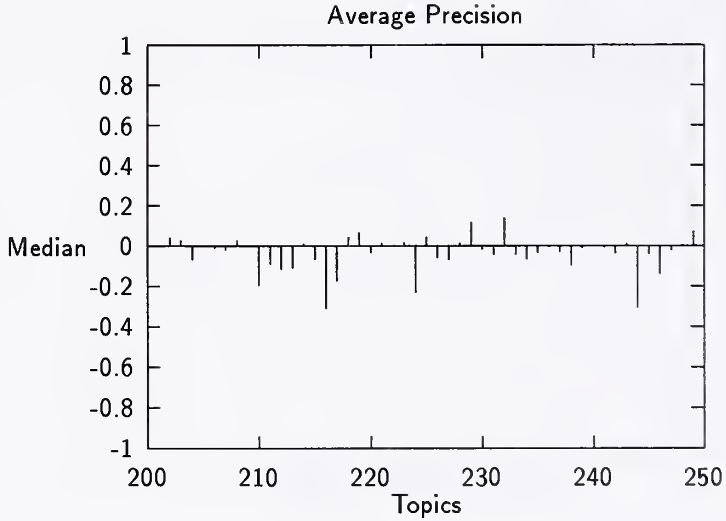
Document Level Averages	
	Precision
At 5 docs	0.4286
At 10 docs	0.3918
At 15 docs	0.3619
At 20 docs	0.3276
At 30 docs	0.2939
At 100 docs	0.1802
At 200 docs	0.1315
At 500 docs	0.0770
At 1000 docs	0.0509

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2088

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00026
0.20	0.00081
0.30	0.00187
0.40	0.00312
0.50	0.00623
0.60	0.01023
0.70	0.02015
0.80	0.05135
0.90	1.96735
1.00	1.00000







Summary Statistics	
Run Number	nyuge4–category A, manual
Number of Topics	49

Total number of documents over all topics	
Retrieved:	48982
Relevant:	6501
Rel_ret:	3536

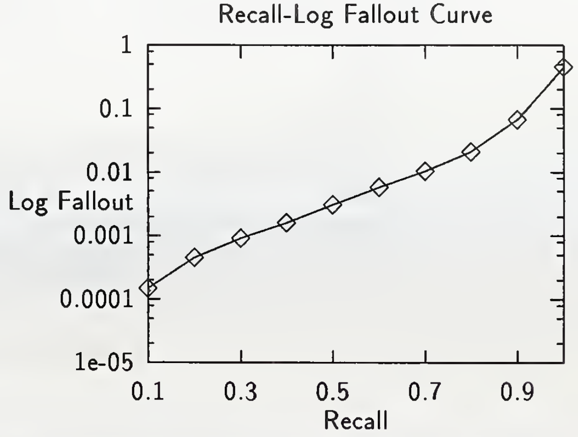
Recall Level Precision Averages	
Recall	Precision
0.00	0.7377
0.10	0.5130
0.20	0.4022
0.30	0.3304
0.40	0.2756
0.50	0.1982
0.60	0.1363
0.70	0.0944
0.80	0.0558
0.90	0.0201
1.00	0.0034

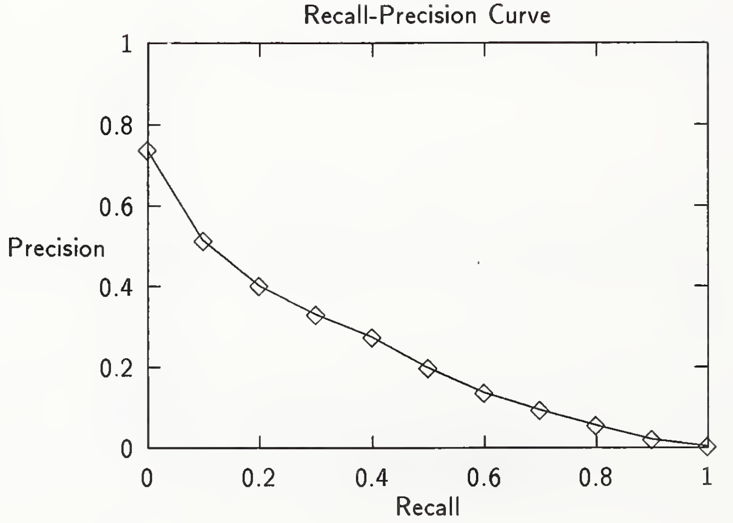
Average precision over all relevant docs	
non-interpolated	0.2272

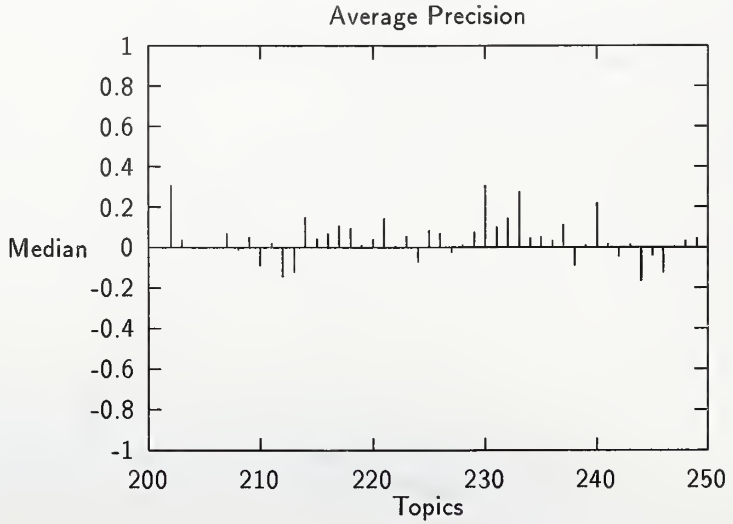
Document Level Averages	
	Precision
At 5 docs	0.5469
At 10 docs	0.4735
At 15 docs	0.4354
At 20 docs	0.4163
At 30 docs	0.3735
At 100 docs	0.2545
At 200 docs	0.1912
At 500 docs	0.1125
At 1000 docs	0.0722

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2780

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00015
0.20	0.00046
0.30	0.00093
0.40	0.00161
0.50	0.00310
0.60	0.00582
0.70	0.01028
0.80	0.02073
0.90	0.06718
1.00	0.44883







Summary Statistics	
Run Number	padreA–category A, automatic
Number of Topics	49

Total number of documents over all topics	
Retrieved:	29615
Relevant:	6501
Rel_ret:	2283

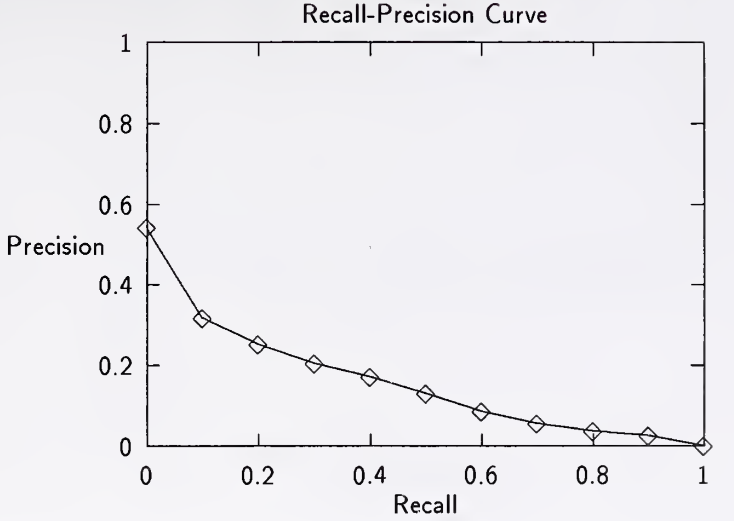
Recall Level Precision Averages	
Recall	Precision
0.00	0.5418
0.10	0.3186
0.20	0.2521
0.30	0.2053
0.40	0.1722
0.50	0.1300
0.60	0.0865
0.70	0.0560
0.80	0.0359
0.90	0.0259
1.00	0.0000

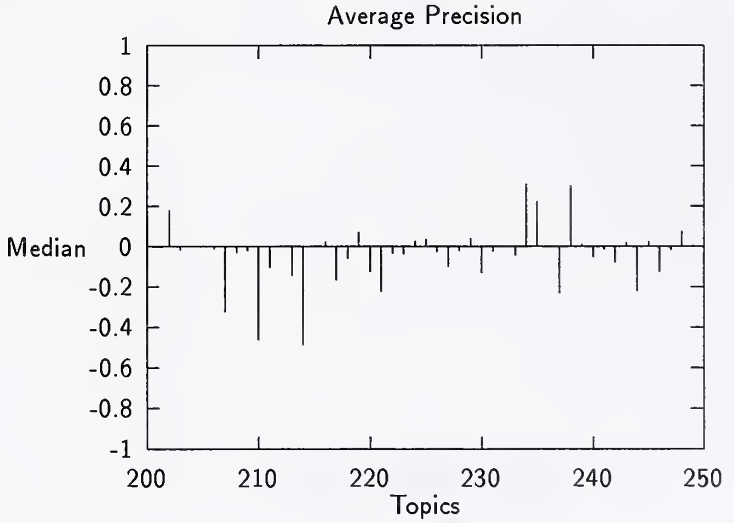
Average precision over all relevant docs	
non-interpolated	0.1453

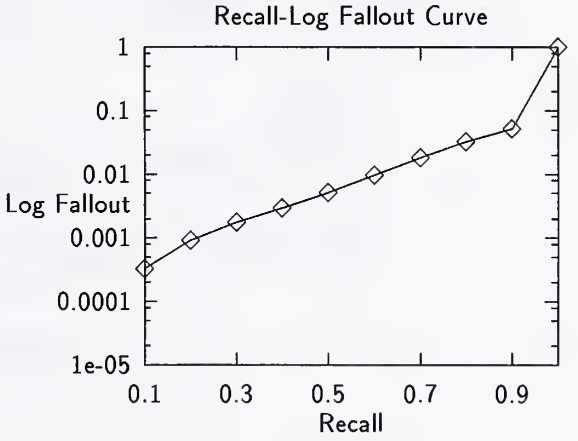
Document Level Averages	
	Precision
At 5 docs	0.3143
At 10 docs	0.2959
At 15 docs	0.2993
At 20 docs	0.2857
At 30 docs	0.2694
At 100 docs	0.1992
At 200 docs	0.1505
At 500 docs	0.0803
At 1000 docs	0.0466

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1954

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00033
0.20	0.00091
0.30	0.00178
0.40	0.00294
0.50	0.00512
0.60	0.00970
0.70	0.01807
0.80	0.03290
0.90	0.05183
1.00	1.00000







Summary Statistics	
Run Number	padreZ–category A, manual
Number of Topics	49

Total number of documents over all topics	
Retrieved:	31304
Relevant:	6501
Rel_ret:	3602

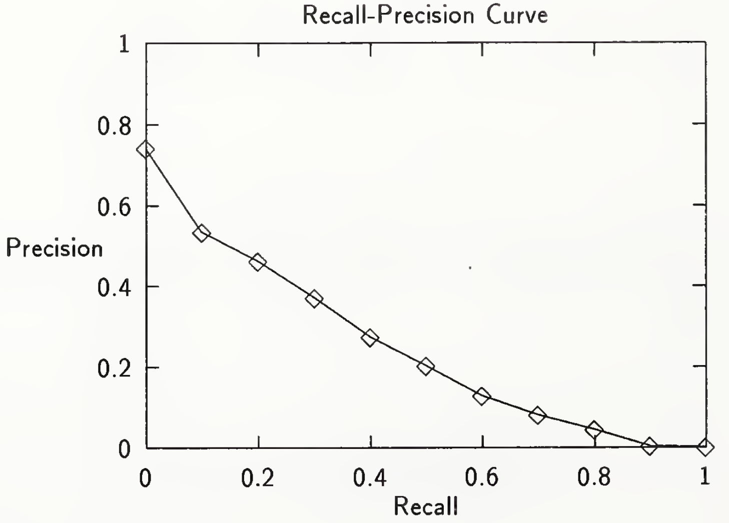
Recall Level Precision Averages	
Recall	Precision
0.00	0.7416
0.10	0.5342
0.20	0.4630
0.30	0.3719
0.40	0.2753
0.50	0.2042
0.60	0.1287
0.70	0.0818
0.80	0.0445
0.90	0.0026
1.00	0.0000

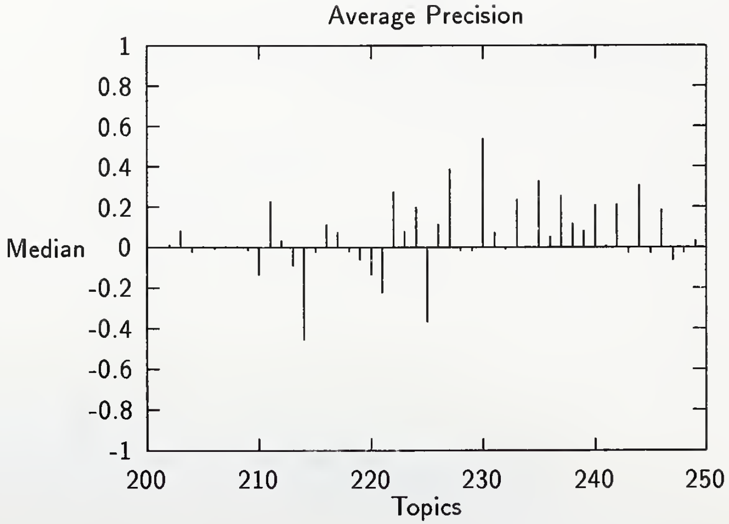
Average precision over all relevant docs	
non-interpolated	0.2383

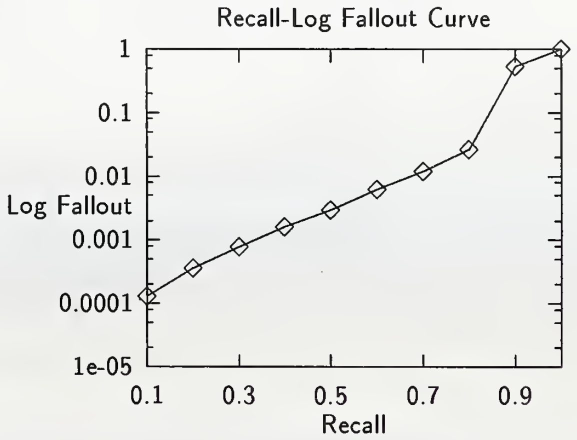
Document Level Averages	
	Precision
At 5 docs	0.5306
At 10 docs	0.5143
At 15 docs	0.4884
At 20 docs	0.4694
At 30 docs	0.4306
At 100 docs	0.2988
At 200 docs	0.2152
At 500 docs	0.1242
At 1000 docs	0.0735

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2934

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00013
0.20	0.00036
0.30	0.00078
0.40	0.00161
0.50	0.00298
0.60	0.00622
0.70	0.01203
0.80	0.02630
0.90	0.52866
1.00	1.00000





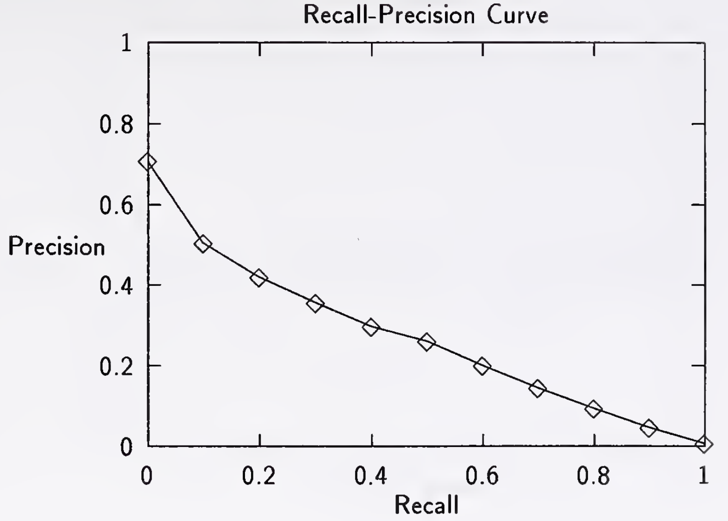


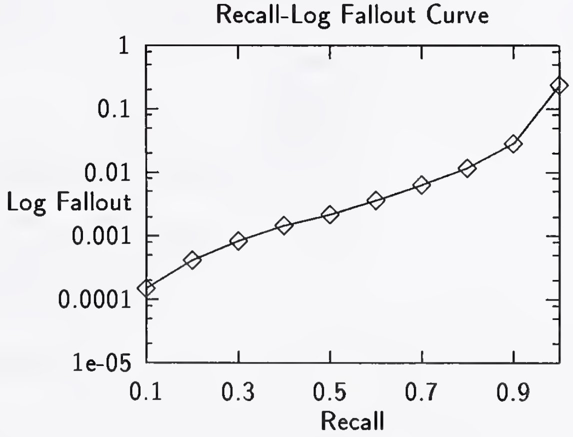
Summary Statistics	
Run Number	pircs1–category A, automatic
Number of Topics	49
Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	3896

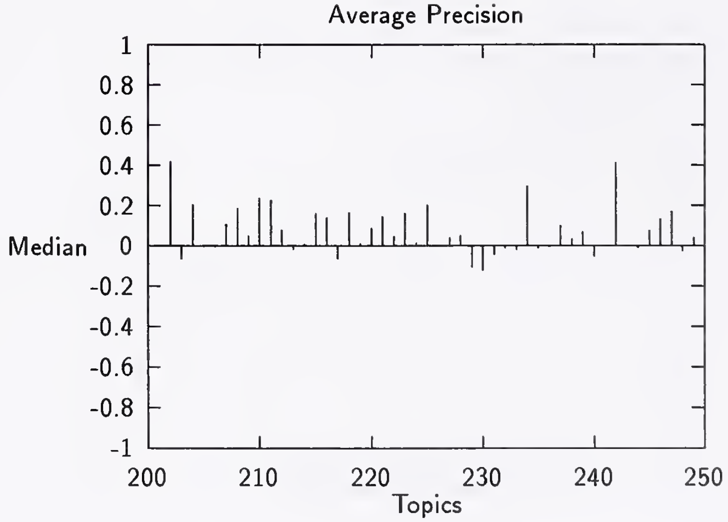
Recall Level Precision Averages	
Recall	Precision
0.00	0.7079
0.10	0.5045
0.20	0.4197
0.30	0.3558
0.40	0.2967
0.50	0.2613
0.60	0.2000
0.70	0.1454
0.80	0.0944
0.90	0.0466
1.00	0.0064
Average precision over all relevant docs	
non-interpolated	0.2599

Document Level Averages	
	Precision
At 5 docs	0.5265
At 10 docs	0.4776
At 15 docs	0.4558
At 20 docs	0.4378
At 30 docs	0.4048
At 100 docs	0.2865
At 200 docs	0.2141
At 500 docs	0.1268
At 1000 docs	0.0795
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3014

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00015
0.20	0.00042
0.30	0.00083
0.40	0.00145
0.50	0.00216
0.60	0.00367
0.70	0.00630
0.80	0.01175
0.90	0.02820
1.00	0.23772







Summary Statistics	
Run Number	pircs2–category A, manual
Number of Topics	49

Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	4562

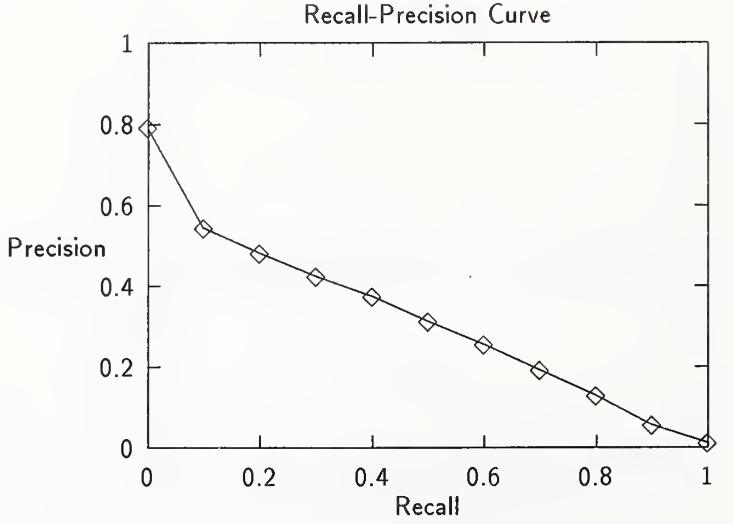
Recall Level Precision Averages	
Recall	Precision
0.00	0.7919
0.10	0.5439
0.20	0.4819
0.30	0.4249
0.40	0.3751
0.50	0.3139
0.60	0.2557
0.70	0.1921
0.80	0.1280
0.90	0.0564
1.00	0.0107

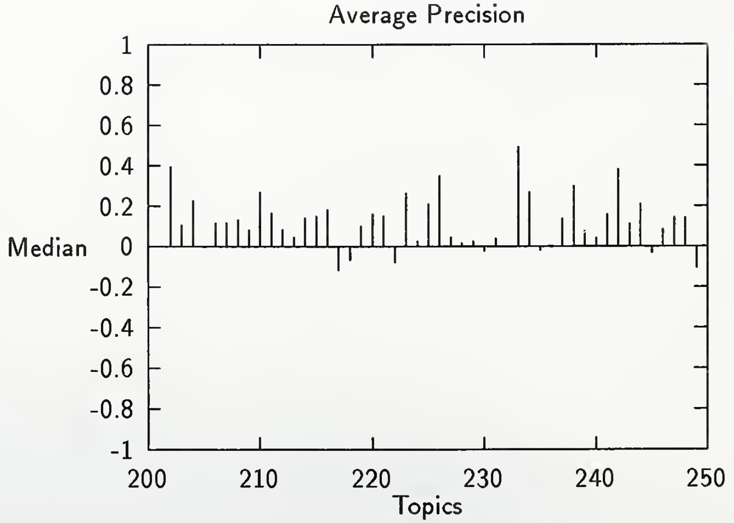
Average precision over all relevant docs	
non-interpolated	0.3064

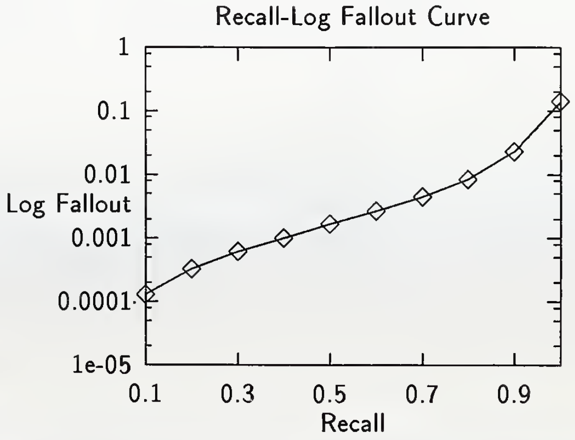
Document Level Averages	
	Precision
At 5 docs	0.5551
At 10 docs	0.5224
At 15 docs	0.4980
At 20 docs	0.4796
At 30 docs	0.4524
At 100 docs	0.3316
At 200 docs	0.2508
At 500 docs	0.1495
At 1000 docs	0.0931

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3421

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00013
0.20	0.00033
0.30	0.00062
0.40	0.00102
0.50	0.00167
0.60	0.00267
0.70	0.00451
0.80	0.00835
0.90	0.02306
1.00	0.14157


Recall-Precision Curve


Average Precision


Recall-Log Fallout Curve

Summary Statistics	
Run Number	siems1–category A, automatic
Number of Topics	49

Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	3790

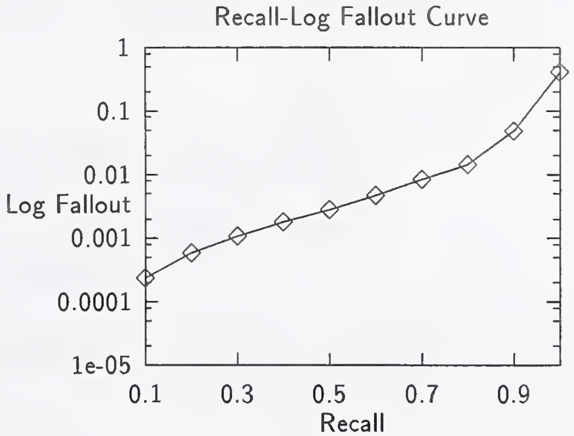
Recall Level Precision Averages	
Recall	Precision
0.00	0.5453
0.10	0.3888
0.20	0.3419
0.30	0.2961
0.40	0.2536
0.50	0.2124
0.60	0.1649
0.70	0.1128
0.80	0.0788
0.90	0.0278
1.00	0.0037

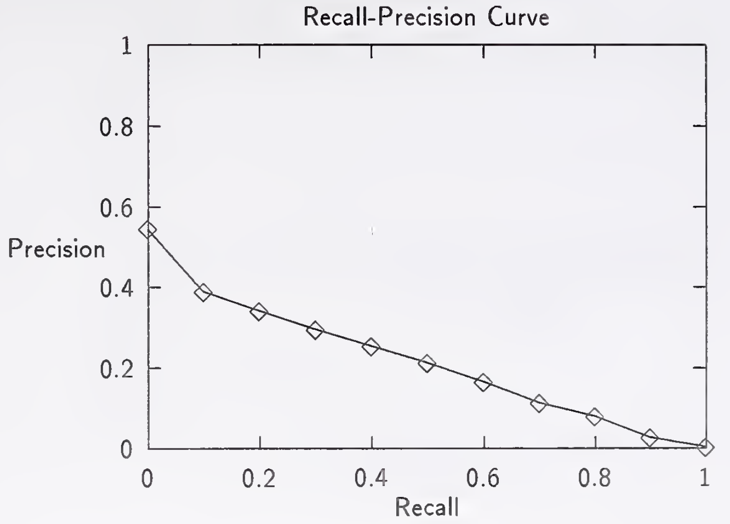
Average precision over all relevant docs	
non-interpolated	0.2031

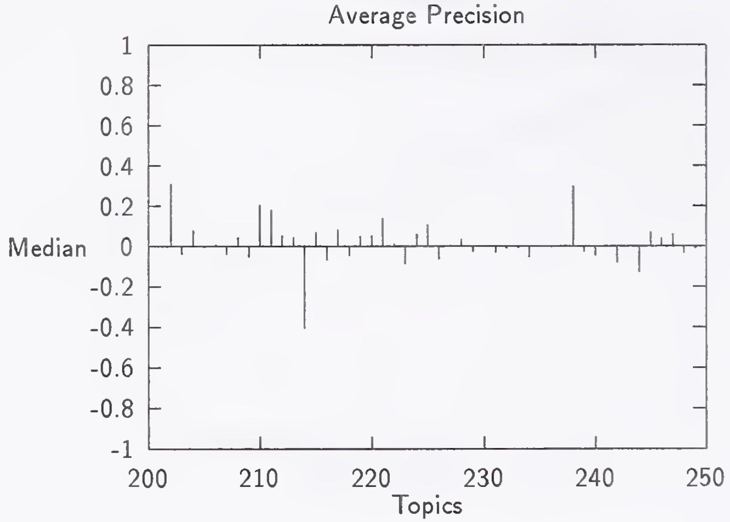
Document Level Averages	
	Precision
At 5 docs	0.3633
At 10 docs	0.3429
At 15 docs	0.3333
At 20 docs	0.3265
At 30 docs	0.3116
At 100 docs	0.2465
At 200 docs	0.1907
At 500 docs	0.1203
At 1000 docs	0.0773

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2566

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00024
0.20	0.00059
0.30	0.00109
0.40	0.00180
0.50	0.00284
0.60	0.00465
0.70	0.00843
0.80	0.01432
0.90	0.04819
1.00	0.41232







Summary Statistics	
Run Number	uwgcl1–category A, manual
Number of Topics	49

Total number of documents over all topics	
Retrieved:	42930
Relevant:	6501
Rel_ret:	4359

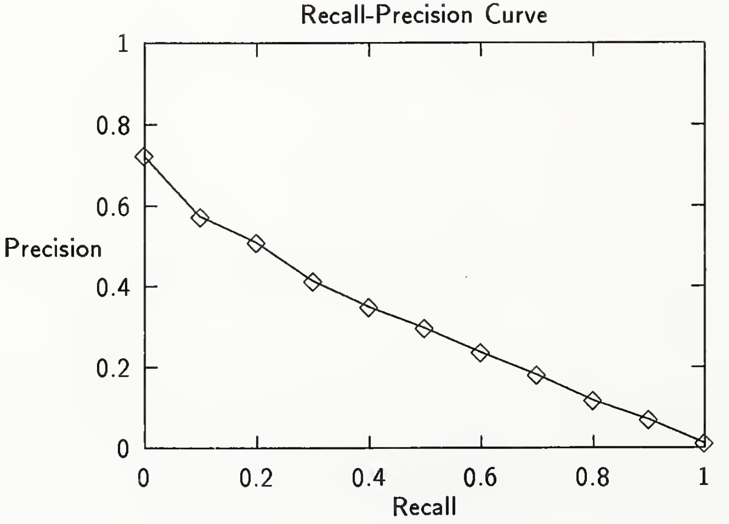
Recall Level Precision Averages	
Recall	Precision
0.00	0.7236
0.10	0.5725
0.20	0.5085
0.30	0.4140
0.40	0.3497
0.50	0.2968
0.60	0.2362
0.70	0.1814
0.80	0.1177
0.90	0.0704
1.00	0.0109

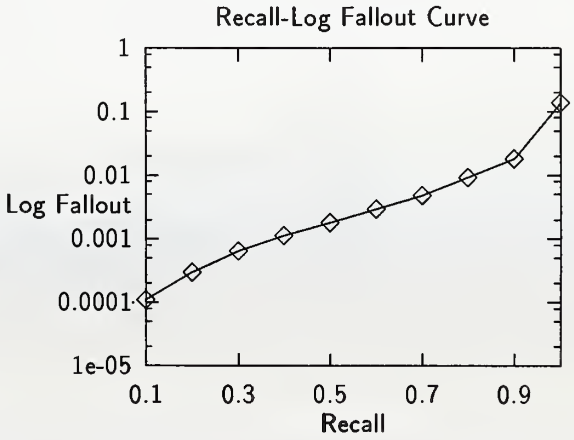
Average precision over all relevant docs	
non-interpolated	0.2994

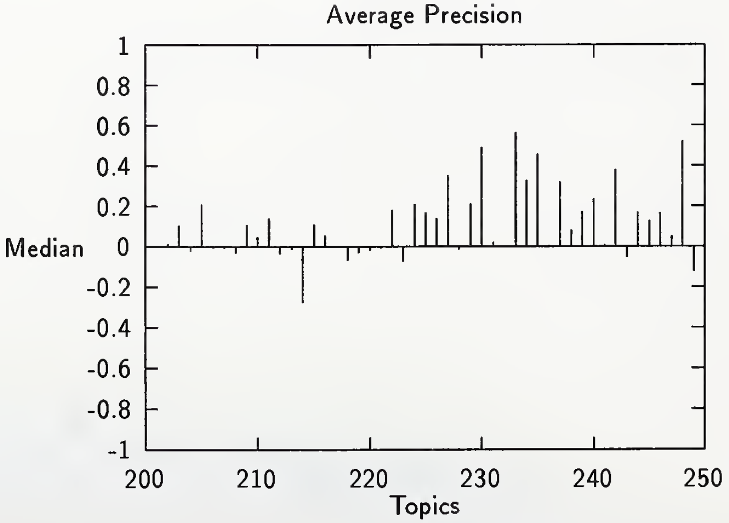
Document Level Averages	
	Precision
At 5 docs	0.5510
At 10 docs	0.5041
At 15 docs	0.4871
At 20 docs	0.4765
At 30 docs	0.4517
At 100 docs	0.3408
At 200 docs	0.2527
At 500 docs	0.1458
At 1000 docs	0.0890

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3347

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00011
0.20	0.00030
0.30	0.00065
0.40	0.00114
0.50	0.00181
0.60	0.00297
0.70	0.00484
0.80	0.00918
0.90	0.01820
1.00	0.13895





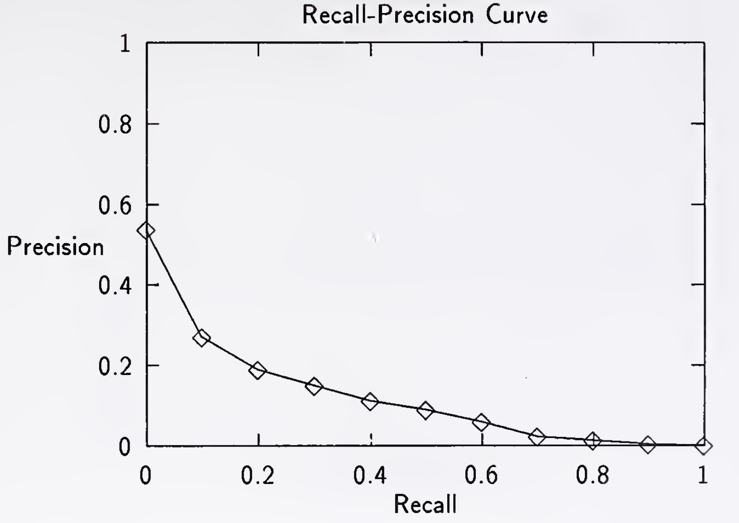


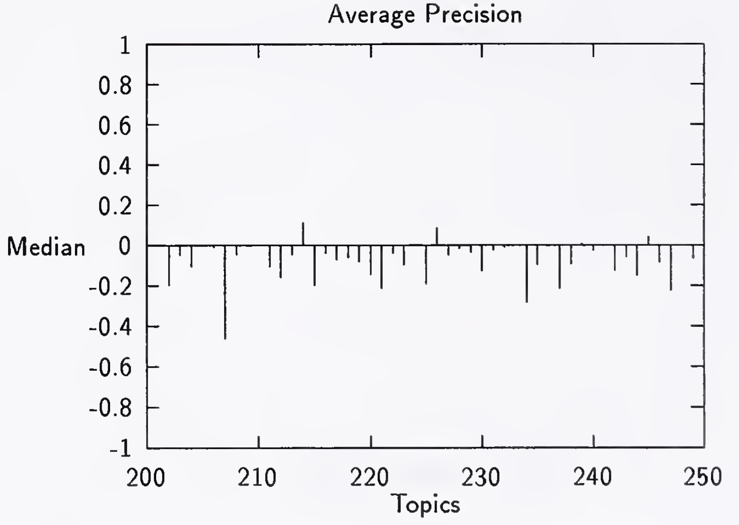
Summary Statistics	
Run Number	virtu4–category A, automatic
Number of Topics	49
Total number of documents over all topics	
Retrieved:	45826
Relevant:	6501
Rel_ret:	2018

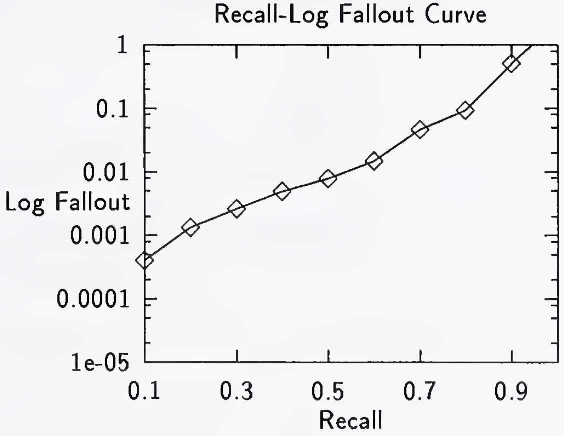
Recall Level Precision Averages	
Recall	Precision
0.00	0.5374
0.10	0.2710
0.20	0.1886
0.30	0.1490
0.40	0.1114
0.50	0.0895
0.60	0.0585
0.70	0.0225
0.80	0.0130
0.90	0.0027
1.00	0.0007
Average precision over all relevant docs	
non-interpolated	0.1082

Document Level Averages	
	Precision
At 5 docs	0.3510
At 10 docs	0.3143
At 15 docs	0.2816
At 20 docs	0.2480
At 30 docs	0.2163
At 100 docs	0.1351
At 200 docs	0.0993
At 500 docs	0.0620
At 1000 docs	0.0412
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1539

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00041
0.20	0.00132
0.30	0.00262
0.40	0.00489
0.50	0.00779
0.60	0.01479
0.70	0.04657
0.80	0.09300
0.90	0.50903
1.00	2.18594





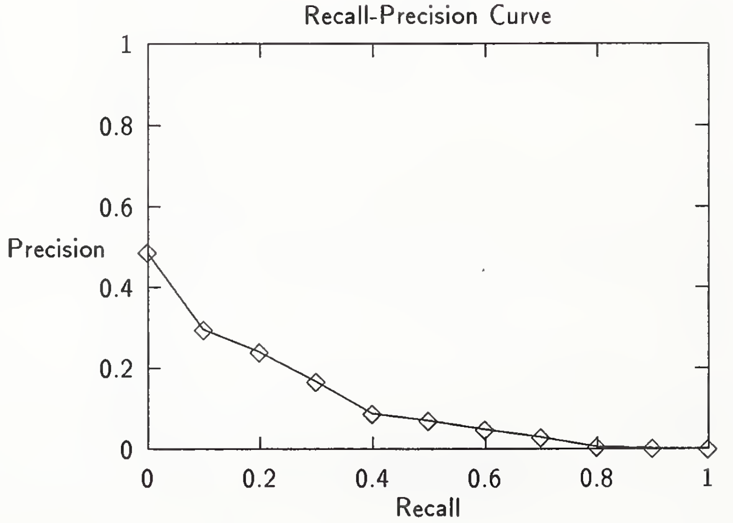


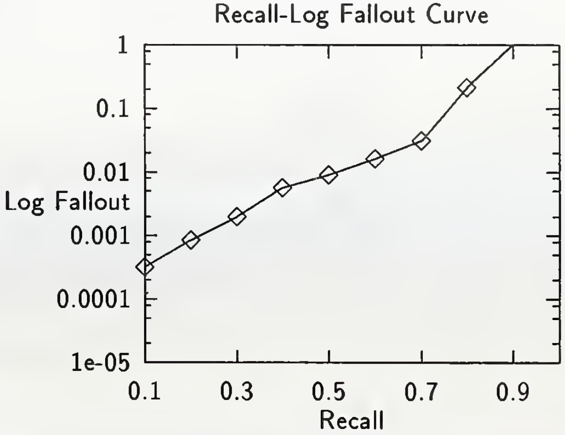
Summary Statistics	
Run Number	drift1–category B, automatic
Number of Topics	48
Total number of documents over all topics	
Retrieved:	48000
Relevant:	2480
Rel_ret:	1350

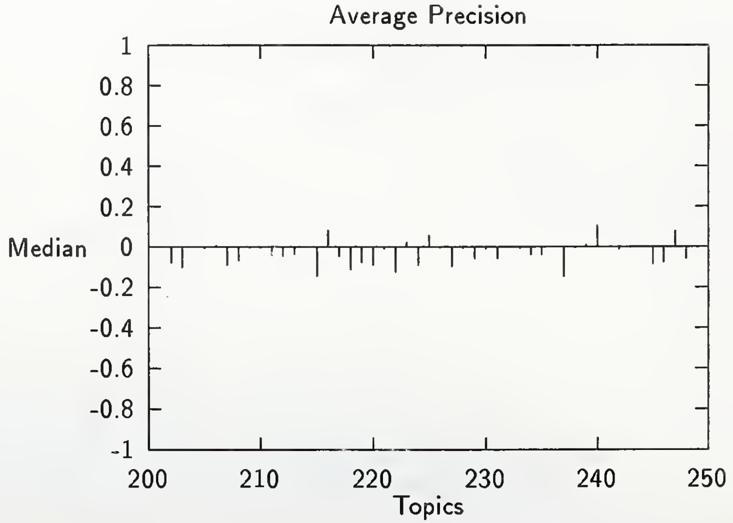
Recall Level Precision Averages	
Recall	Precision
0.00	0.4859
0.10	0.2952
0.20	0.2399
0.30	0.1669
0.40	0.0870
0.50	0.0693
0.60	0.0473
0.70	0.0289
0.80	0.0049
0.90	0.0012
1.00	0.0003
Average precision over all relevant docs	
non-interpolated	0.1108

Document Level Averages	
	Precision
At 5 docs	0.2833
At 10 docs	0.2375
At 15 docs	0.2097
At 20 docs	0.1885
At 30 docs	0.1639
At 100 docs	0.1044
At 200 docs	0.0746
At 500 docs	0.0443
At 1000 docs	0.0281
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1396

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00032
0.20	0.00085
0.30	0.00200
0.40	0.00560
0.50	0.00897
0.60	0.01613
0.70	0.03140
0.80	0.21691
0.90	1.00013
1.00	4.44904







Summary Statistics	
Run Number	drift2–category B, automatic
Number of Topics	48

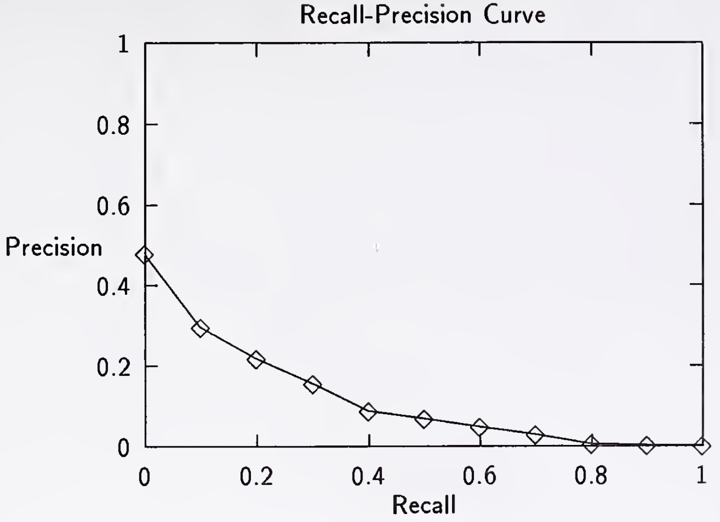
Total number of documents over all topics	
Retrieved:	48000
Relevant:	2480
Rel_ret:	1352

Recall Level Precision Averages	
Recall	Precision
0.00	0.4787
0.10	0.2964
0.20	0.2171
0.30	0.1564
0.40	0.0874
0.50	0.0677
0.60	0.0470
0.70	0.0285
0.80	0.0054
0.90	0.0019
1.00	0.0004

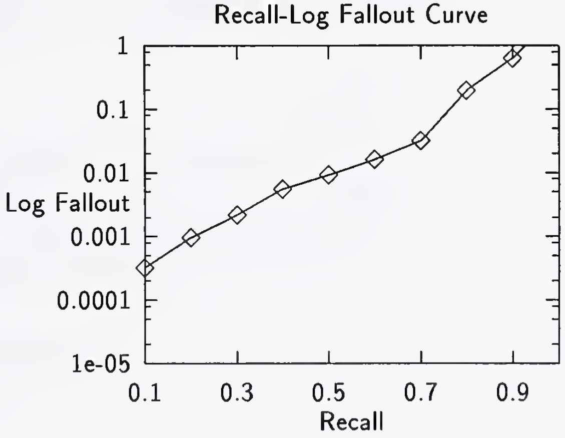
Average precision over all relevant docs	
non-interpolated	0.1073

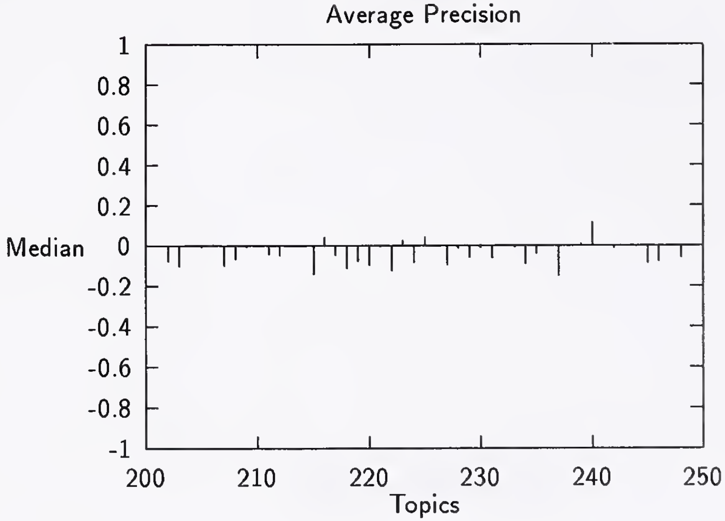
Document Level Averages	
	Precision
At 5 docs	0.2750
At 10 docs	0.2333
At 15 docs	0.2014
At 20 docs	0.1854
At 30 docs	0.1694
At 100 docs	0.1062
At 200 docs	0.0750
At 500 docs	0.0441
At 1000 docs	0.0282

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1386



Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00032
0.20	0.00096
0.30	0.00216
0.40	0.00558
0.50	0.00919
0.60	0.01624
0.70	0.03186
0.80	0.19673
0.90	0.63122
1.00	3.33645





Summary Statistics	
Run Number	glair1–category B, automatic
Number of Topics	48

Total number of documents over all topics	
Retrieved:	48000
Relevant:	2480
Rel_ret:	1703

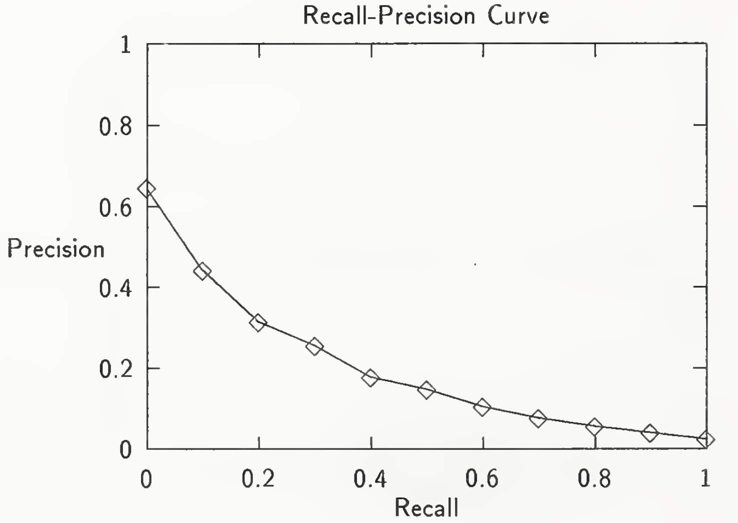
Recall Level Precision Averages	
Recall	Precision
0.00	0.6453
0.10	0.4423
0.20	0.3149
0.30	0.2559
0.40	0.1783
0.50	0.1483
0.60	0.1057
0.70	0.0760
0.80	0.0554
0.90	0.0401
1.00	0.0233

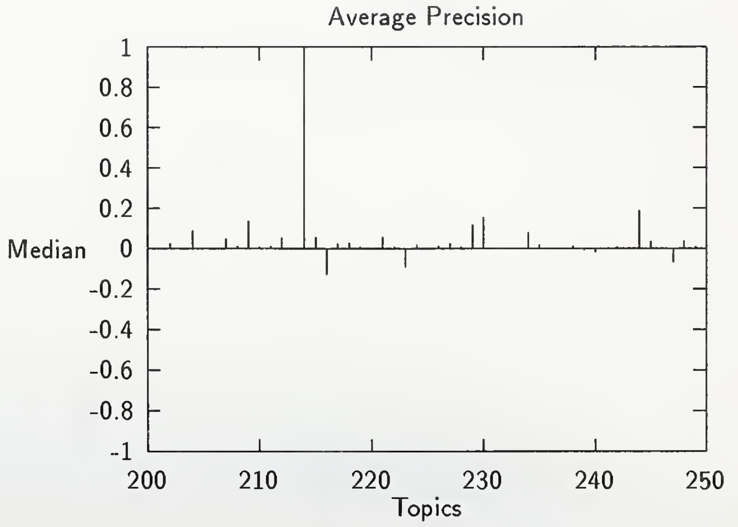
Average precision over all relevant docs	
non-interpolated	0.1857

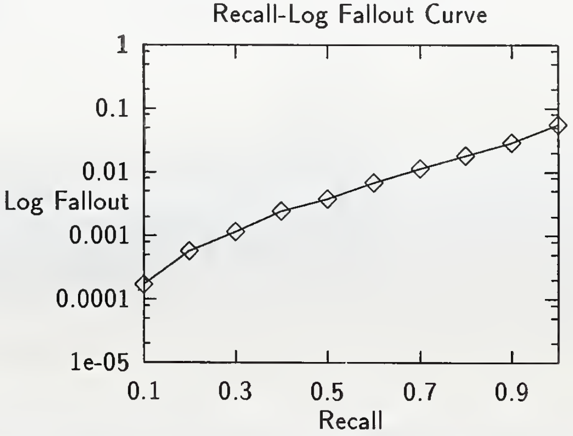
Document Level Averages	
	Precision
At 5 docs	0.3708
At 10 docs	0.3187
At 15 docs	0.2917
At 20 docs	0.2667
At 30 docs	0.2361
At 100 docs	0.1452
At 200 docs	0.0985
At 500 docs	0.0578
At 1000 docs	0.0355

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2216

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00017
0.20	0.00058
0.30	0.00116
0.40	0.00246
0.50	0.00383
0.60	0.00678
0.70	0.01136
0.80	0.01821
0.90	0.02876
1.00	0.05597





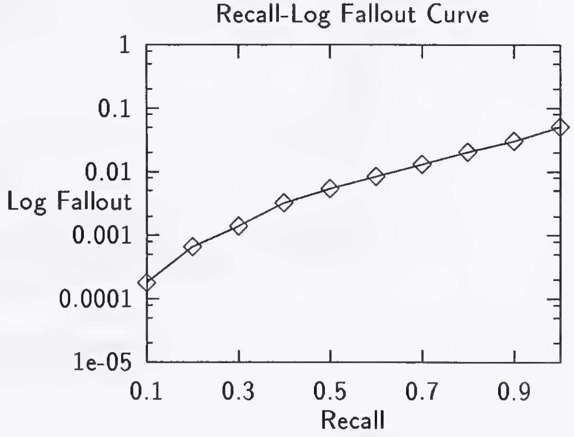


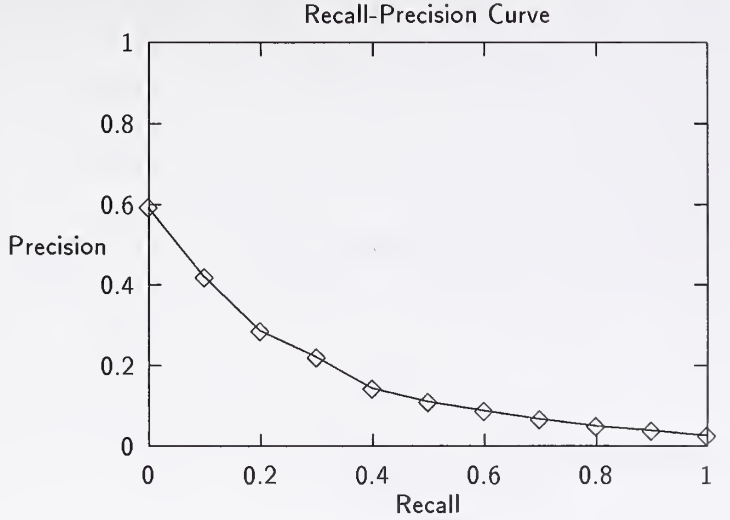
Summary Statistics	
Run Number	KU1–category B, automatic
Number of Topics	48
Total number of documents over all topics	
Retrieved:	48000
Relevant:	2480
Rel_ret:	1514

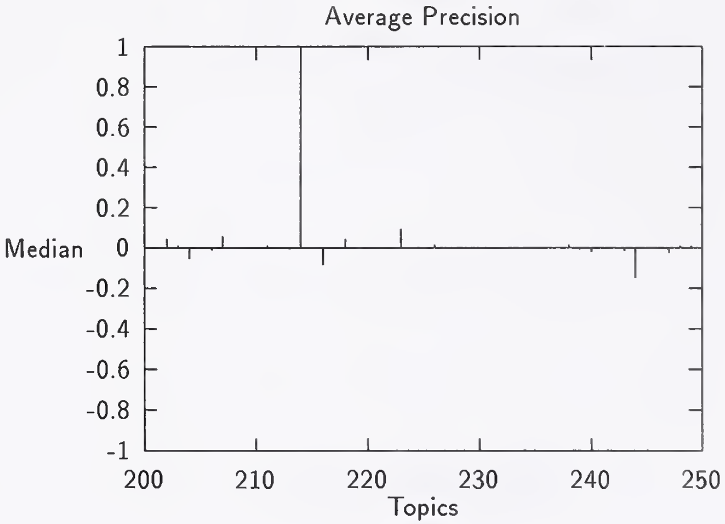
Recall Level Precision Averages	
Recall	Precision
0.00	0.5934
0.10	0.4196
0.20	0.2865
0.30	0.2216
0.40	0.1428
0.50	0.1096
0.60	0.0876
0.70	0.0664
0.80	0.0497
0.90	0.0380
1.00	0.0259
Average precision over all relevant docs	
non-interpolated	0.1638

Document Level Averages	
	Precision
At 5 docs	0.3458
At 10 docs	0.2771
At 15 docs	0.2639
At 20 docs	0.2542
At 30 docs	0.2194
At 100 docs	0.1279
At 200 docs	0.0895
At 500 docs	0.0516
At 1000 docs	0.0315
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2002

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00018
0.20	0.00066
0.30	0.00141
0.40	0.00321
0.50	0.00542
0.60	0.00834
0.70	0.01314
0.80	0.02042
0.90	0.03042
1.00	0.05021





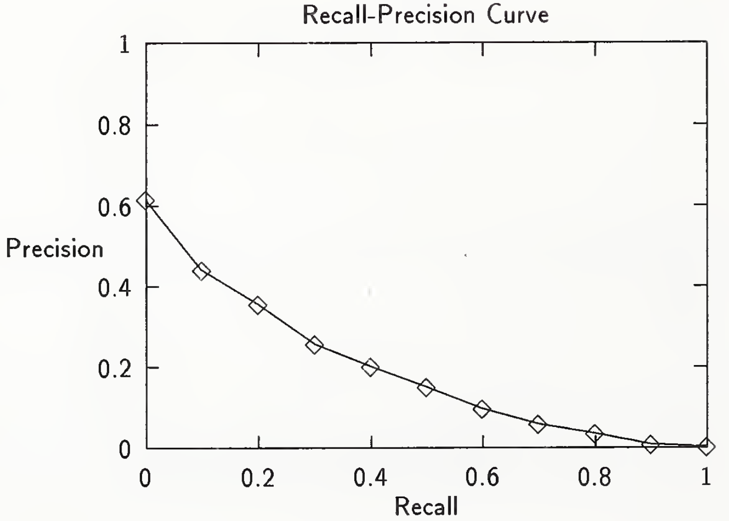


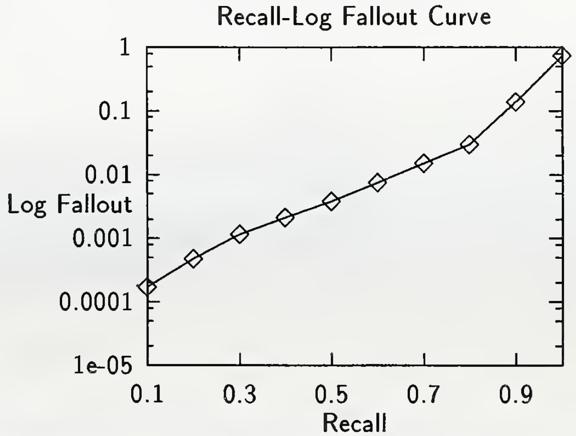
Summary Statistics	
Run Number	UniNE3–category B, automatic
Number of Topics	48
Total number of documents over all topics	
Retrieved:	48000
Relevant:	2480
Rel_ret:	1653

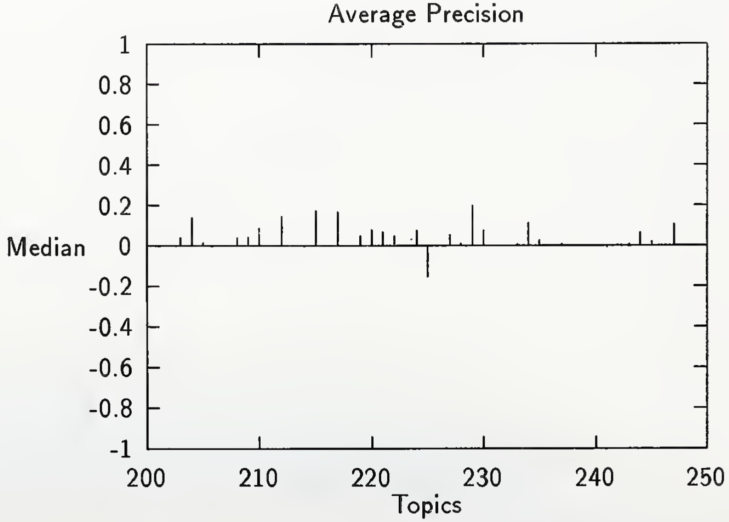
Recall Level Precision Averages	
Recall	Precision
0.00	0.6153
0.10	0.4400
0.20	0.3557
0.30	0.2574
0.40	0.2017
0.50	0.1505
0.60	0.0974
0.70	0.0587
0.80	0.0346
0.90	0.0087
1.00	0.0018
Average precision over all relevant docs	
non-interpolated	0.1797

Document Level Averages	
	Precision
At 5 docs	0.3500
At 10 docs	0.3229
At 15 docs	0.2944
At 20 docs	0.2802
At 30 docs	0.2458
At 100 docs	0.1508
At 200 docs	0.1070
At 500 docs	0.0582
At 1000 docs	0.0344
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2301

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00017
0.20	0.00048
0.30	0.00116
0.40	0.00211
0.50	0.00377
0.60	0.00742
0.70	0.01499
0.80	0.02980
0.90	0.13691
1.00	0.74039







Summary Statistics	
Run Number	UniNE4–category B, automatic
Number of Topics	48
Total number of documents over all topics	
Retrieved:	48000
Relevant:	2480
Rel_ret:	1543

Recall Level Precision Averages	
Recall	Precision
0.00	0.5754
0.10	0.4282
0.20	0.3431
0.30	0.2450
0.40	0.1893
0.50	0.1417
0.60	0.0894
0.70	0.0516
0.80	0.0293
0.90	0.0087
1.00	0.0018

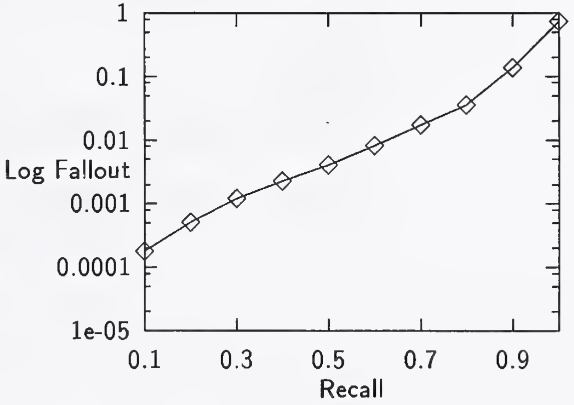
Average precision over all relevant docs	
non-interpolated	0.1714

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00018
0.20	0.00051
0.30	0.00123
0.40	0.00229
0.50	0.00404
0.60	0.00816
0.70	0.01718
0.80	0.03539
0.90	0.13691
1.00	0.74039

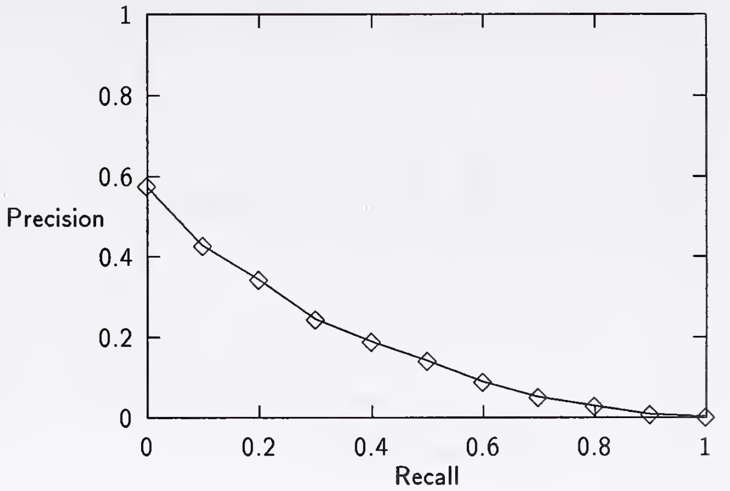
Document Level Averages	
	Precision
At 5 docs	0.3375
At 10 docs	0.3083
At 15 docs	0.2861
At 20 docs	0.2719
At 30 docs	0.2368
At 100 docs	0.1396
At 200 docs	0.0984
At 500 docs	0.0531
At 1000 docs	0.0321

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2203

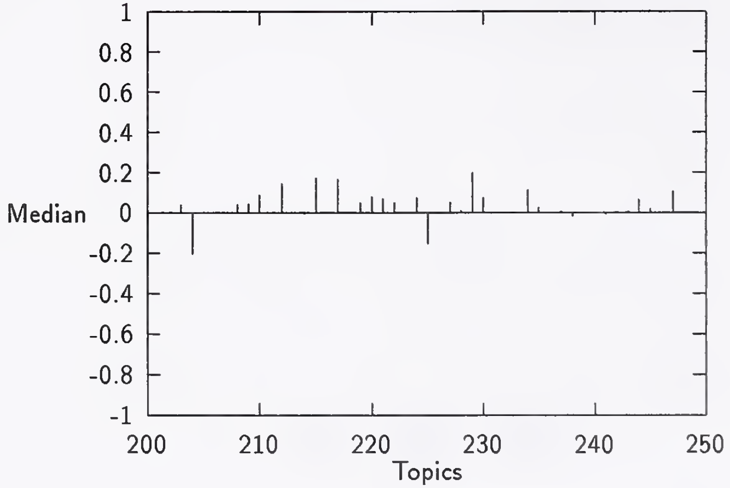
Recall-Log Fallout Curve



Recall-Precision Curve



Average Precision

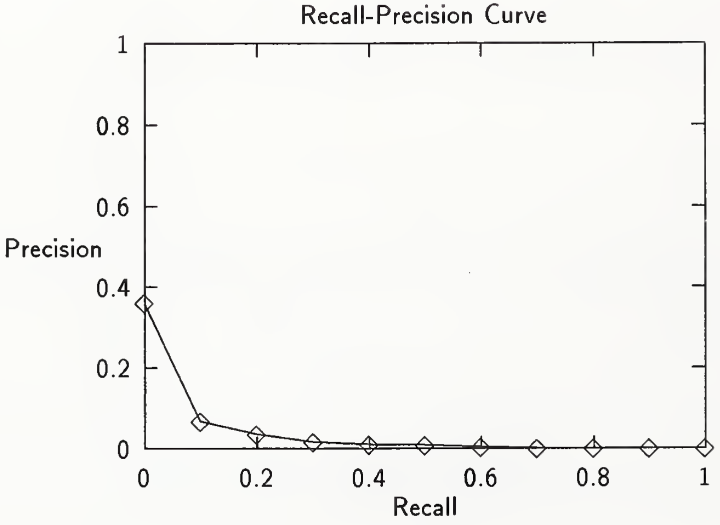


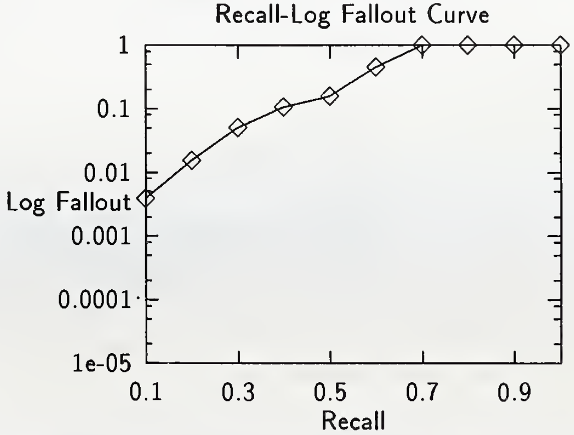
Summary Statistics	
Run Number	ACQROU–category A, manual
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	6576
Rel_ret:	1331

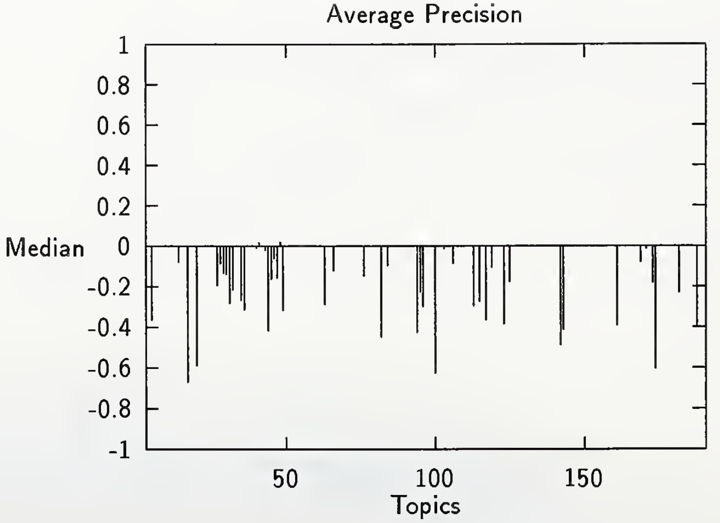
Recall Level Precision Averages	
Recall	Precision
0.00	0.3613
0.10	0.0662
0.20	0.0342
0.30	0.0160
0.40	0.0103
0.50	0.0087
0.60	0.0036
0.70	0.0000
0.80	0.0000
0.90	0.0000
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.0232

Document Level Averages	
	Precision
At 5 docs	0.1520
At 10 docs	0.1280
At 15 docs	0.1187
At 20 docs	0.1070
At 30 docs	0.0953
At 100 docs	0.0730
At 200 docs	0.0533
At 500 docs	0.0365
At 1000 docs	0.0266
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0558

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00389
0.20	0.01557
0.30	0.05085
0.40	0.10594
0.50	0.15703
0.60	0.45774
0.70	1.00000
0.80	1.00000
0.90	1.00000
1.00	1.00000





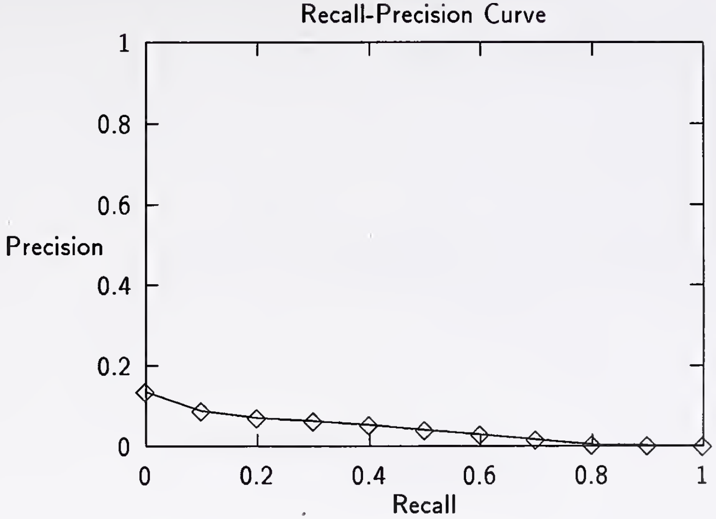


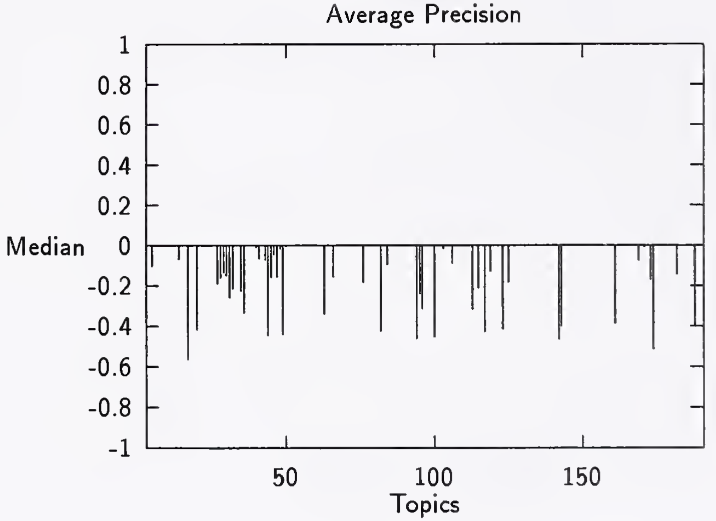
Summary Statistics	
Run Number	Brkly11–category A, automatic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	6576
Rel_ret:	2151

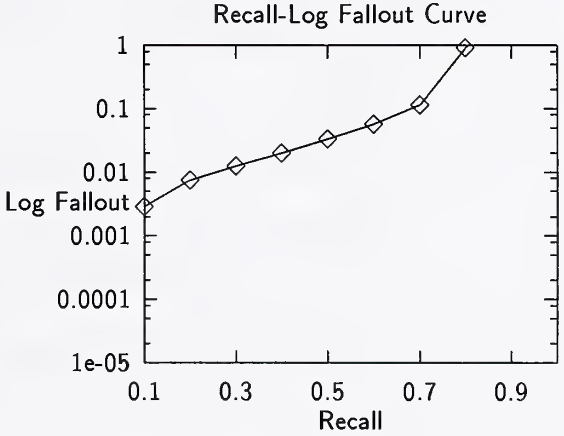
Recall Level Precision Averages	
Recall	Precision
0.00	0.1354
0.10	0.0870
0.20	0.0692
0.30	0.0623
0.40	0.0523
0.50	0.0398
0.60	0.0281
0.70	0.0165
0.80	0.0024
0.90	0.0009
1.00	0.0001
Average precision over all relevant docs	
non-interpolated	0.0345

Document Level Averages	
	Precision
At 5 docs	0.0200
At 10 docs	0.0220
At 15 docs	0.0307
At 20 docs	0.0350
At 30 docs	0.0387
At 100 docs	0.0430
At 200 docs	0.0535
At 500 docs	0.0518
At 1000 docs	0.0430
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0475

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00289
0.20	0.00742
0.30	0.01245
0.40	0.01998
0.50	0.03325
0.60	0.05720
0.70	0.11501
0.80	0.91657
0.90	2.75386
1.00	27.56063







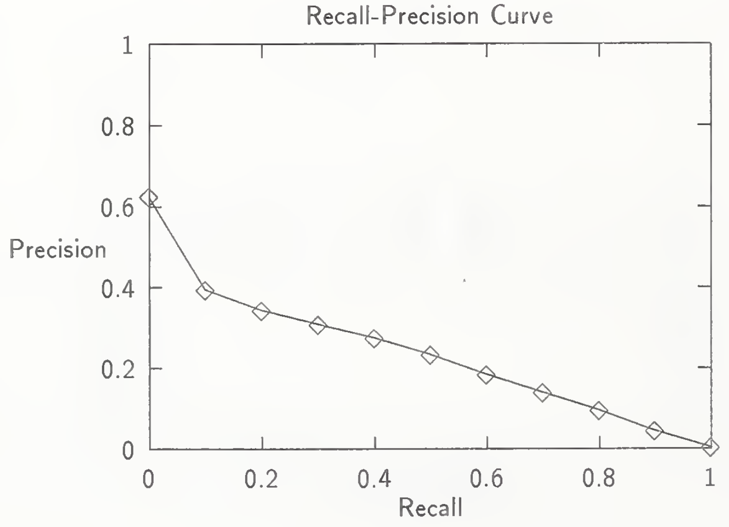
Summary Statistics	
Run Number	Brkly12–category A, automatic
Number of Topics	50

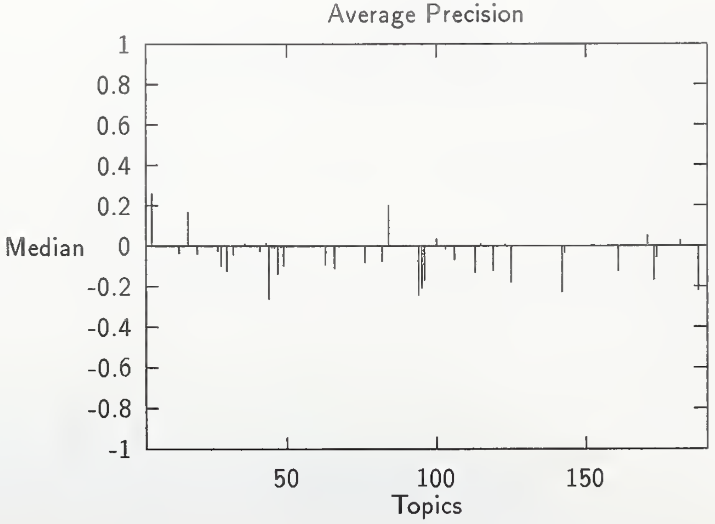
Total number of documents over all topics	
Retrieved:	50000
Relevant:	6576
Rel_ret:	4373

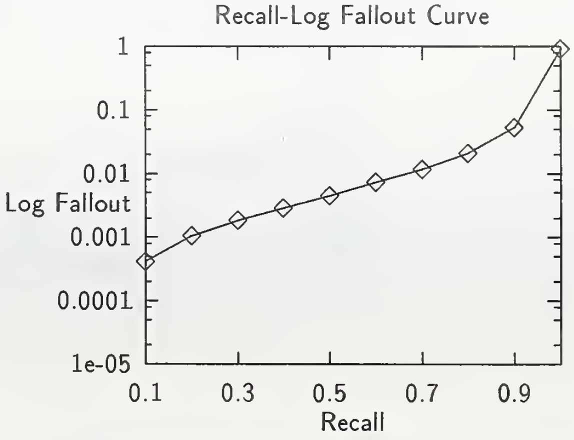
Recall Level Precision Averages	
Recall	Precision
0.00	0.6247
0.10	0.3947
0.20	0.3434
0.30	0.3086
0.40	0.2753
0.50	0.2337
0.60	0.1851
0.70	0.1406
0.80	0.0953
0.90	0.0448
1.00	0.0030
Average precision over all relevant docs	
non-interpolated	0.2163

Document Level Averages	
	Precision
At 5 docs	0.3920
At 10 docs	0.3280
At 15 docs	0.3053
At 20 docs	0.2910
At 30 docs	0.2847
At 100 docs	0.2916
At 200 docs	0.2361
At 500 docs	0.1434
At 1000 docs	0.0875
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2679

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00042
0.20	0.00105
0.30	0.00185
0.40	0.00290
0.50	0.00452
0.60	0.00728
0.70	0.01179
0.80	0.02093
0.90	0.05289
1.00	0.91602







Summary Statistics	
Run Number	cityr1–category A, automatic
Number of Topics	50

Total number of documents over all topics	
Retrieved:	50000
Relevant:	6576
Rel_ret:	5451

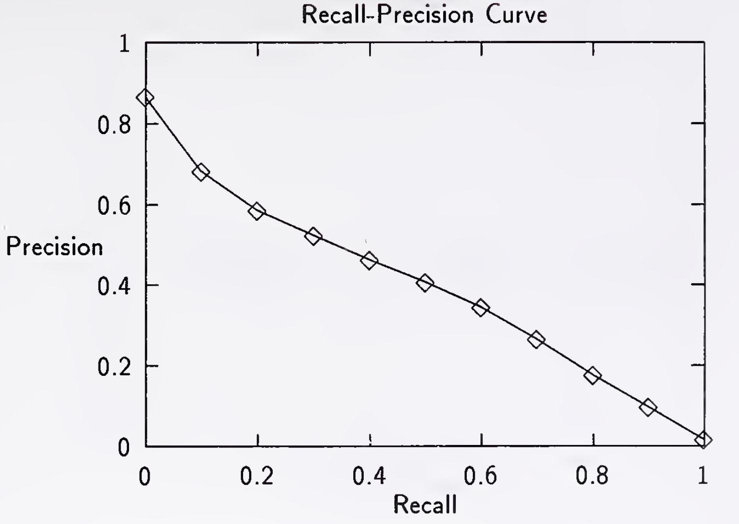
Recall Level Precision Averages	
Recall	Precision
0.00	0.8666
0.10	0.6826
0.20	0.5845
0.30	0.5230
0.40	0.4628
0.50	0.4077
0.60	0.3448
0.70	0.2654
0.80	0.1769
0.90	0.0967
1.00	0.0160

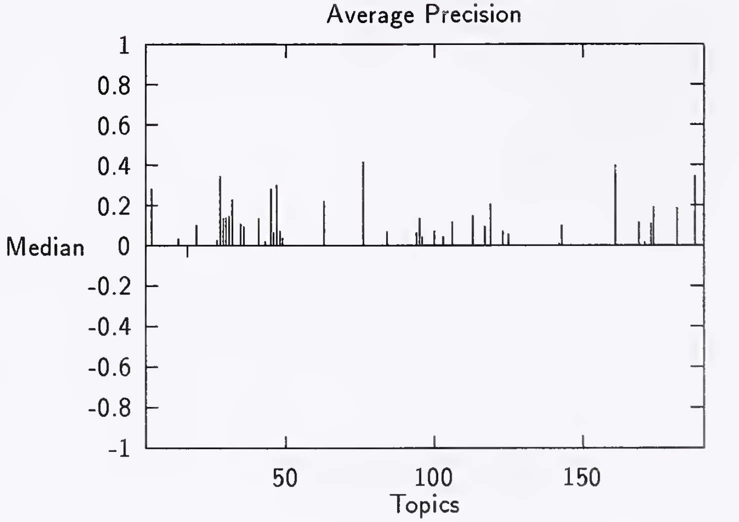
Average precision over all relevant docs	
non-interpolated	0.3868

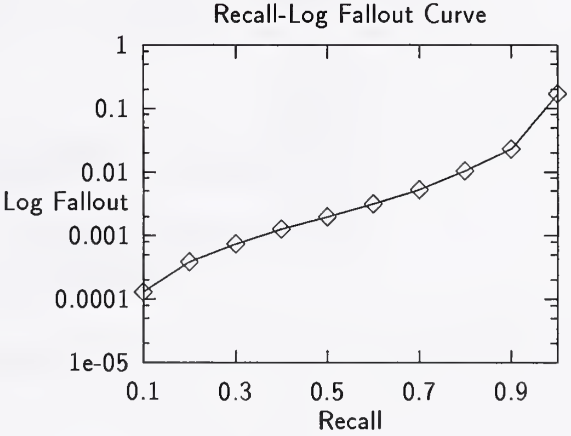
Document Level Averages	
	Precision
At 5 docs	0.6840
At 10 docs	0.6440
At 15 docs	0.6067
At 20 docs	0.5990
At 30 docs	0.5693
At 100 docs	0.4436
At 200 docs	0.3384
At 500 docs	0.1924
At 1000 docs	0.1090

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.4080

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00013
0.20	0.00039
0.30	0.00075
0.40	0.00128
0.50	0.00200
0.60	0.00314
0.70	0.00534
0.80	0.01026
0.90	0.02317
1.00	0.16951







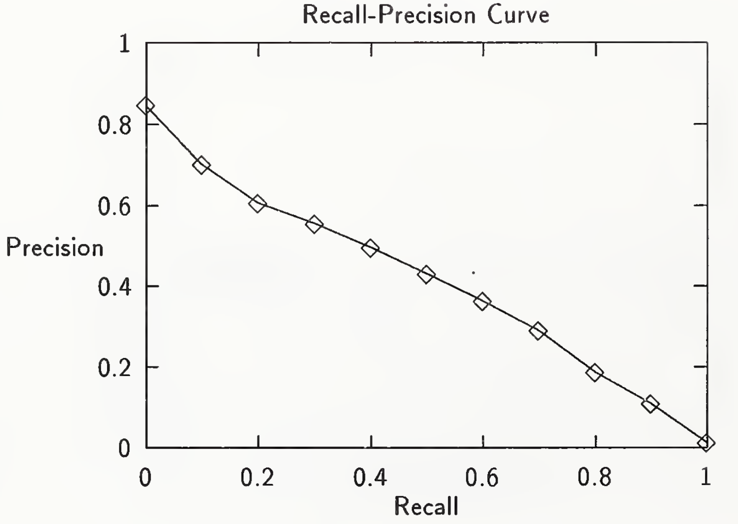
Summary Statistics	
Run Number	cityr2–category A, automatic
Number of Topics	50

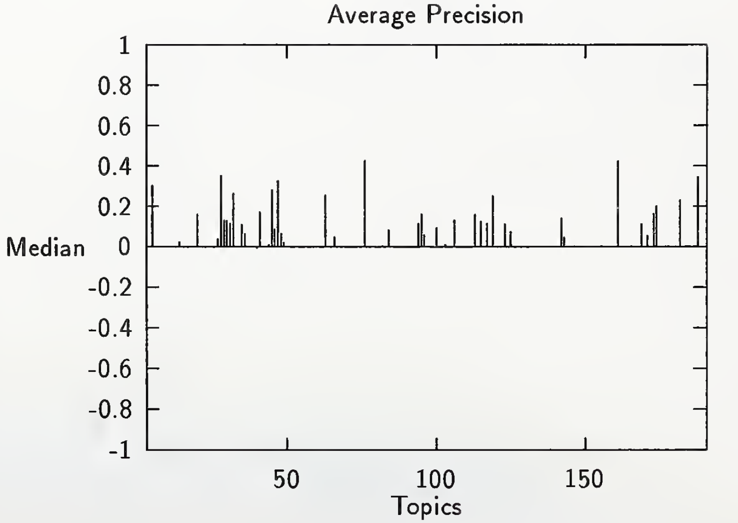
Total number of documents over all topics	
Retrieved:	50000
Relevant:	6576
Rel_ret:	5550

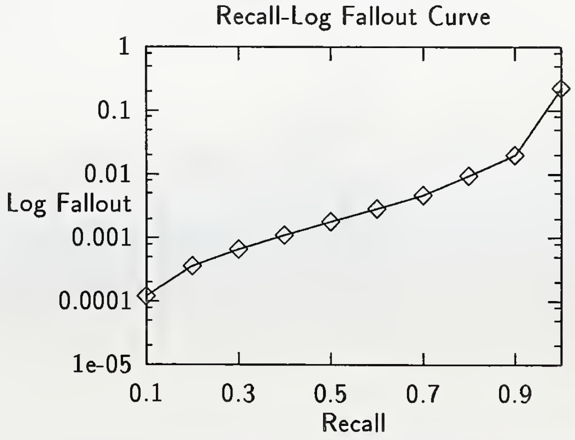
Recall Level Precision Averages	
Recall	Precision
0.00	0.8470
0.10	0.7017
0.20	0.6060
0.30	0.5543
0.40	0.4956
0.50	0.4306
0.60	0.3643
0.70	0.2908
0.80	0.1876
0.90	0.1099
1.00	0.0121
Average precision over all relevant docs	
non-interpolated	0.4074

Document Level Averages	
	Precision
At 5 docs	0.6880
At 10 docs	0.6620
At 15 docs	0.6320
At 20 docs	0.6160
At 30 docs	0.5707
At 100 docs	0.4652
At 200 docs	0.3465
At 500 docs	0.1962
At 1000 docs	0.1110
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.4221

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00012
0.20	0.00036
0.30	0.00066
0.40	0.00112
0.50	0.00182
0.60	0.00289
0.70	0.00471
0.80	0.00955
0.90	0.02009
1.00	0.22504







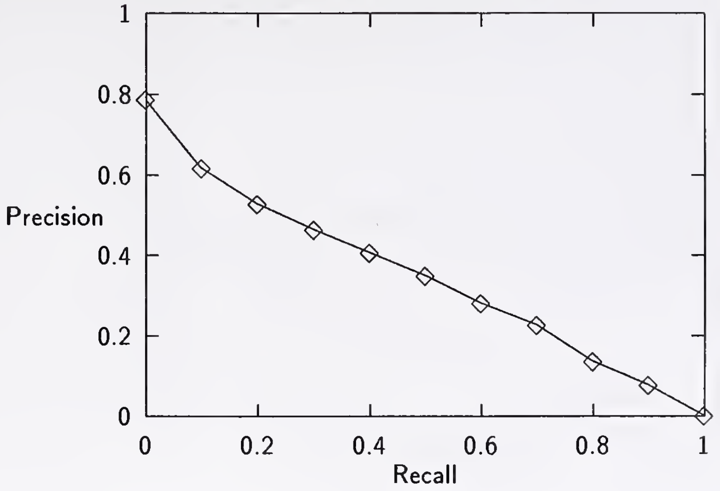
Summary Statistics	
Run Number	CrnlRE–category A, automatic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	6576
Rel_ret:	4917

Recall Level Precision Averages	
Recall	Precision
0.00	0.7874
0.10	0.6167
0.20	0.5285
0.30	0.4647
0.40	0.4072
0.50	0.3496
0.60	0.2820
0.70	0.2270
0.80	0.1367
0.90	0.0779
1.00	0.0019
Average precision over all relevant docs	
non-interpolated	0.3380

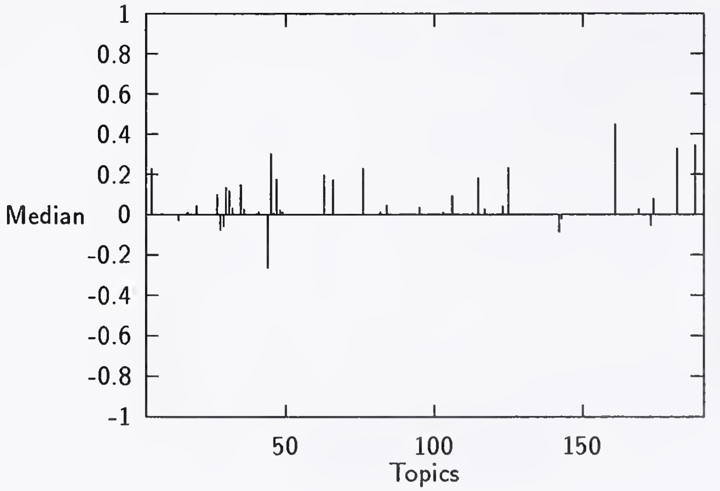
Document Level Averages	
	Precision
At 5 docs	0.6160
At 10 docs	0.6000
At 15 docs	0.5693
At 20 docs	0.5520
At 30 docs	0.5253
At 100 docs	0.4080
At 200 docs	0.3051
At 500 docs	0.1716
At 1000 docs	0.0983
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3758

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00017
0.20	0.00049
0.30	0.00095
0.40	0.00161
0.50	0.00256
0.60	0.00421
0.70	0.00657
0.80	0.01393
0.90	0.02936
1.00	1.44795

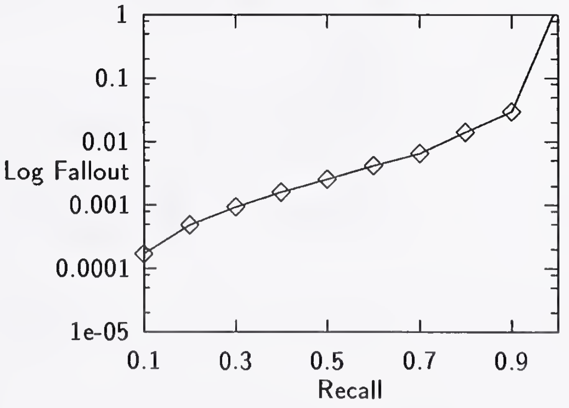
Recall-Precision Curve



Average Precision



Recall-Log Fallout Curve



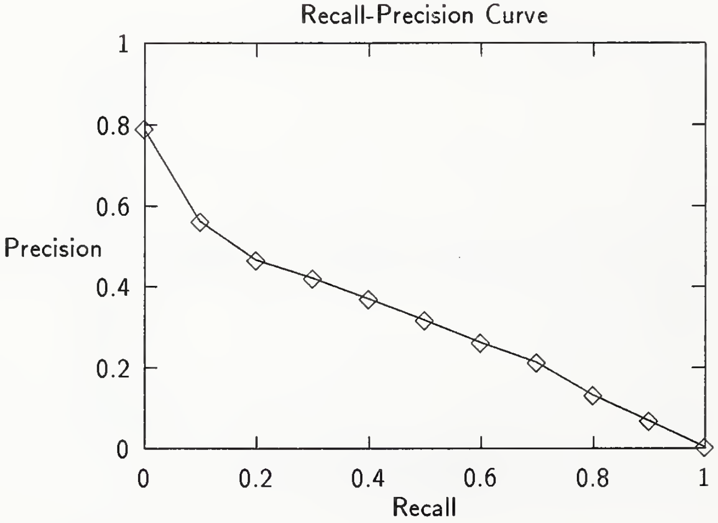
Summary Statistics	
Run Number	CrnIRL–category A, automatic
Number of Topics	50

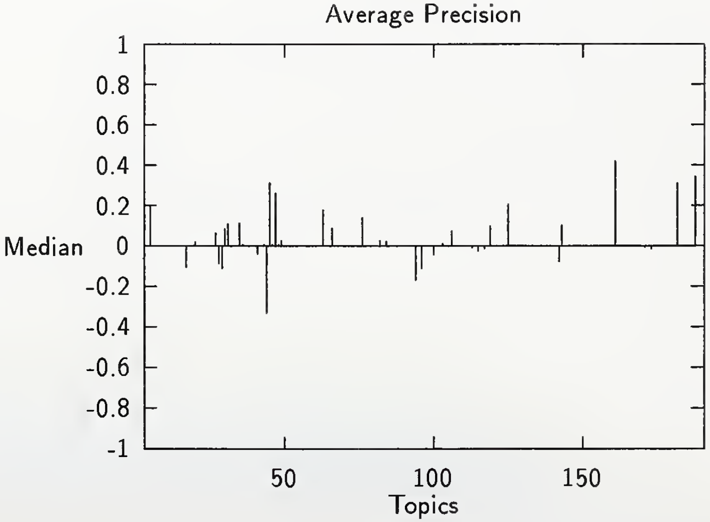
Total number of documents over all topics	
Retrieved:	50000
Relevant:	6576
Rel_ret:	4789

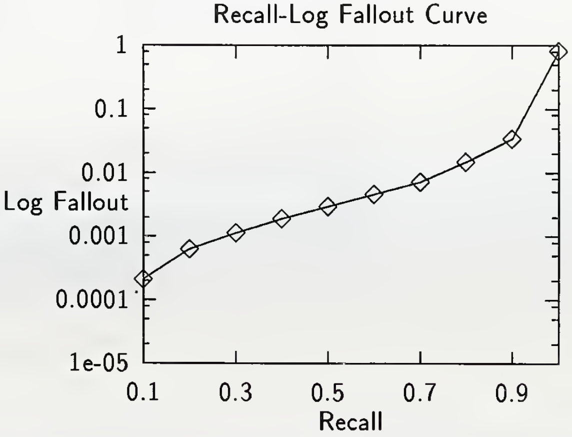
Recall Level Precision Averages	
Recall	Precision
0.00	0.7899
0.10	0.5620
0.20	0.4664
0.30	0.4210
0.40	0.3701
0.50	0.3186
0.60	0.2631
0.70	0.2132
0.80	0.1313
0.90	0.0676
1.00	0.0034
Average precision over all relevant docs	
non-interpolated	0.3112

Document Level Averages	
	Precision
At 5 docs	0.6000
At 10 docs	0.5720
At 15 docs	0.5360
At 20 docs	0.5170
At 30 docs	0.4900
At 100 docs	0.3772
At 200 docs	0.2826
At 500 docs	0.1647
At 1000 docs	0.0958
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3481

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00021
0.20	0.00063
0.30	0.00114
0.40	0.00188
0.50	0.00295
0.60	0.00463
0.70	0.00712
0.80	0.01459
0.90	0.03422
1.00	0.80793







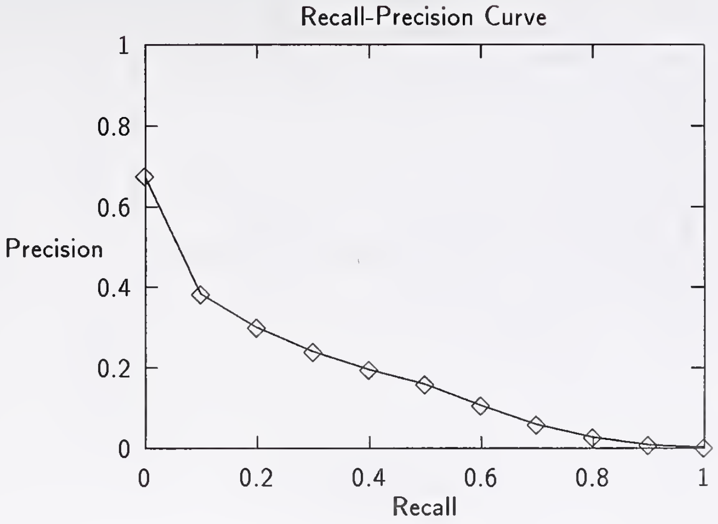
Summary Statistics	
Run Number	HNC11–category A, automatic
Number of Topics	50

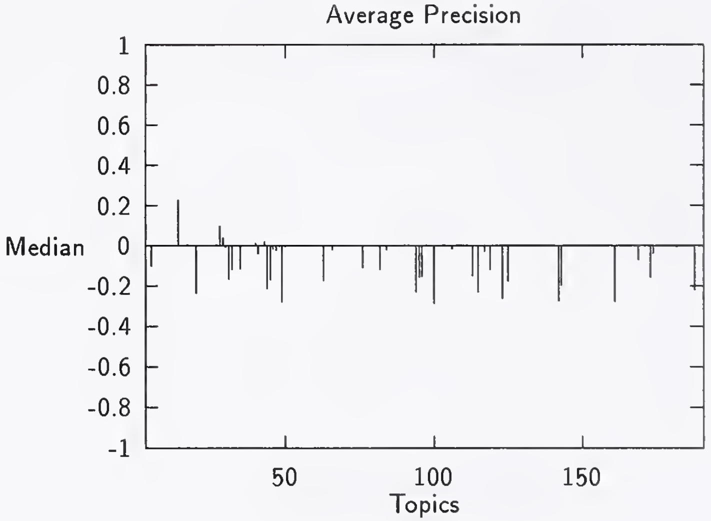
Total number of documents over all topics	
Retrieved:	50000
Relevant:	6576
Rel_ret:	3854

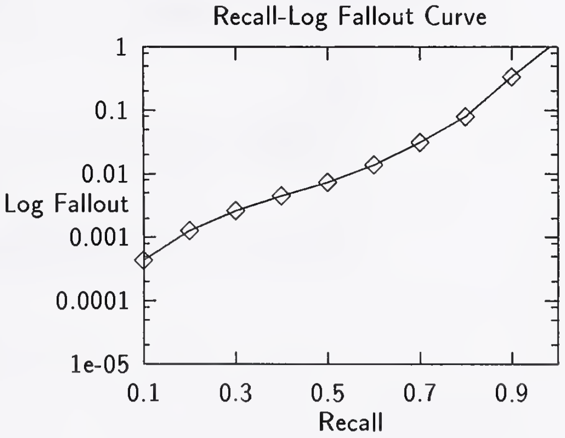
Recall Level Precision Averages	
Recall	Precision
0.00	0.6762
0.10	0.3832
0.20	0.3004
0.30	0.2408
0.40	0.1959
0.50	0.1590
0.60	0.1071
0.70	0.0584
0.80	0.0273
0.90	0.0074
1.00	0.0022
Average precision over all relevant docs	
non-interpolated	0.1727

Document Level Averages	
	Precision
At 5 docs	0.4600
At 10 docs	0.4080
At 15 docs	0.3800
At 20 docs	0.3690
At 30 docs	0.3493
At 100 docs	0.2718
At 200 docs	0.2064
At 500 docs	0.1251
At 1000 docs	0.0771
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2348

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00044
0.20	0.00128
0.30	0.00261
0.40	0.00453
0.50	0.00729
0.60	0.01379
0.70	0.03111
0.80	0.07857
0.90	0.33275
1.00	1.25012







Summary Statistics	
Run Number	HNC21–category A, automatic
Number of Topics	50

Total number of documents over all topics	
Retrieved:	50000
Relevant:	6576
Rel_ret:	3760

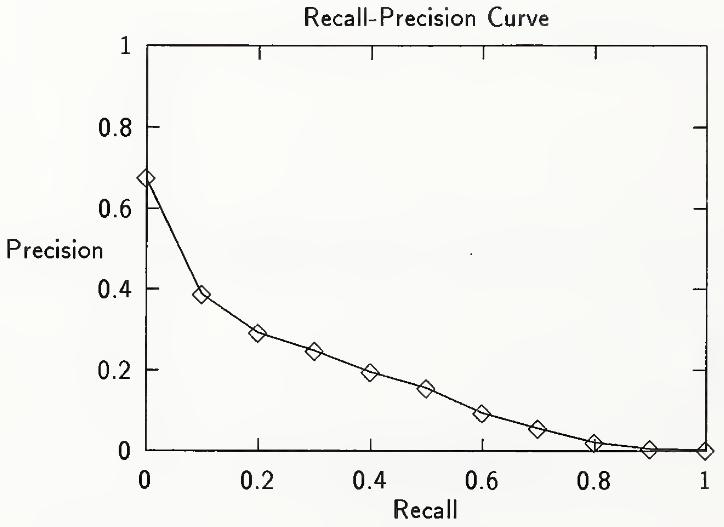
Recall Level Precision Averages	
Recall	Precision
0.00	0.6764
0.10	0.3882
0.20	0.2925
0.30	0.2474
0.40	0.1963
0.50	0.1552
0.60	0.0937
0.70	0.0555
0.80	0.0210
0.90	0.0044
1.00	0.0015

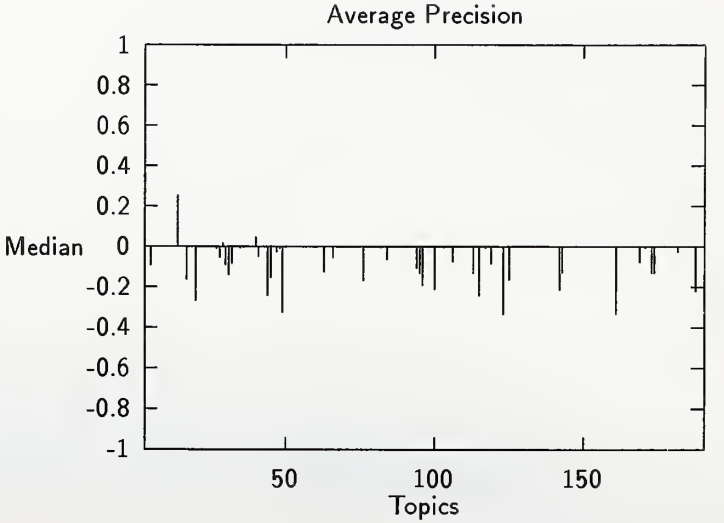
Average precision over all relevant docs	
non-interpolated	0.1693

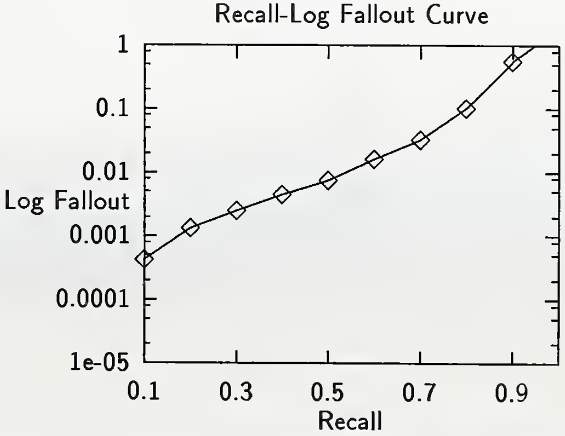
Document Level Averages	
	Precision
At 5 docs	0.4680
At 10 docs	0.4140
At 15 docs	0.3960
At 20 docs	0.3790
At 30 docs	0.3547
At 100 docs	0.2662
At 200 docs	0.2006
At 500 docs	0.1203
At 1000 docs	0.0752

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2274

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00043
0.20	0.00133
0.30	0.00252
0.40	0.00451
0.50	0.00750
0.60	0.01600
0.70	0.03284
0.80	0.10280
0.90	0.56132
1.00	1.83480





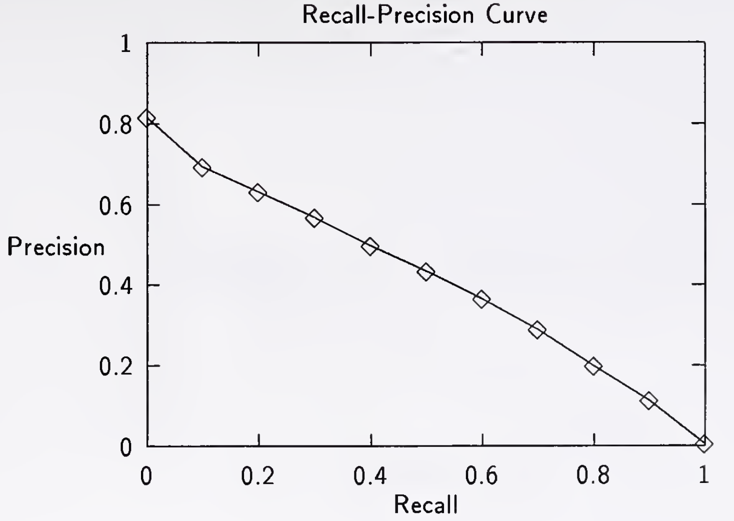


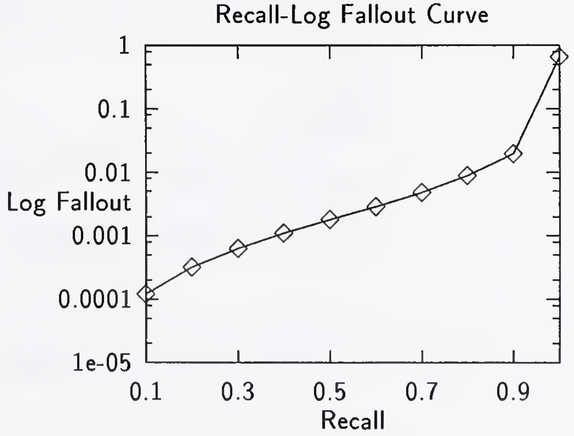
Summary Statistics	
Run Number	INQ203–category A, automatic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	6576
Rel_ret:	5612

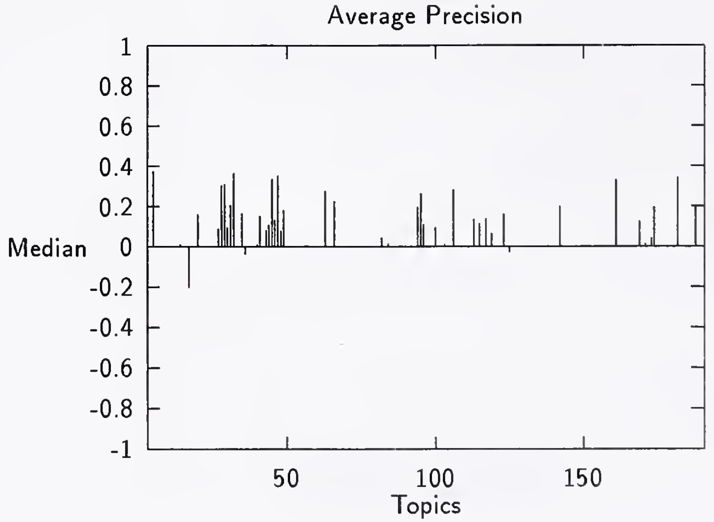
Recall Level Precision Averages	
Recall	Precision
0.00	0.8158
0.10	0.6934
0.20	0.6311
0.30	0.5670
0.40	0.4976
0.50	0.4333
0.60	0.3658
0.70	0.2886
0.80	0.1984
0.90	0.1124
1.00	0.0042
Average precision over all relevant docs	
non-interpolated	0.4103

Document Level Averages	
	Precision
At 5 docs	0.6480
At 10 docs	0.6560
At 15 docs	0.6347
At 20 docs	0.6220
At 30 docs	0.5880
At 100 docs	0.4810
At 200 docs	0.3558
At 500 docs	0.1978
At 1000 docs	0.1122
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.4316

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00012
0.20	0.00032
0.30	0.00063
0.40	0.00111
0.50	0.00180
0.60	0.00287
0.70	0.00476
0.80	0.00891
0.90	0.01959
1.00	0.65351







Summary Statistics	
Run Number	INQ204–category A, automatic
Number of Topics	50

Total number of documents over all topics	
Retrieved:	50000
Relevant:	6576
Rel_ret:	5528

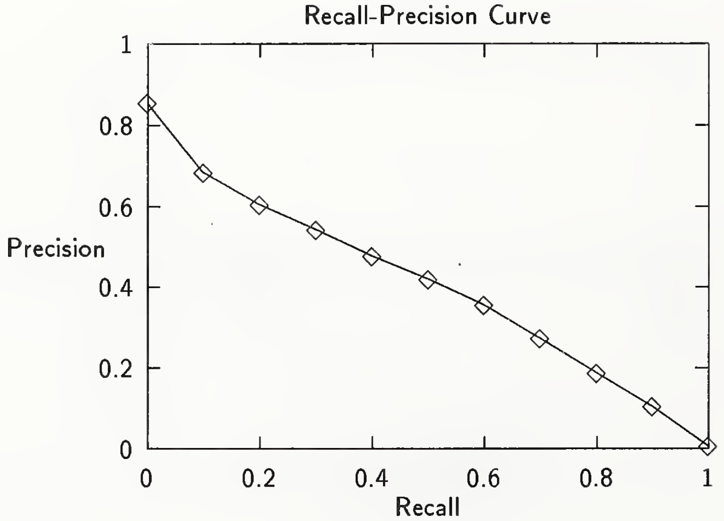
Recall Level Precision Averages	
Recall	Precision
0.00	0.8557
0.10	0.6841
0.20	0.6042
0.30	0.5417
0.40	0.4773
0.50	0.4193
0.60	0.3568
0.70	0.2742
0.80	0.1873
0.90	0.1042
1.00	0.0060

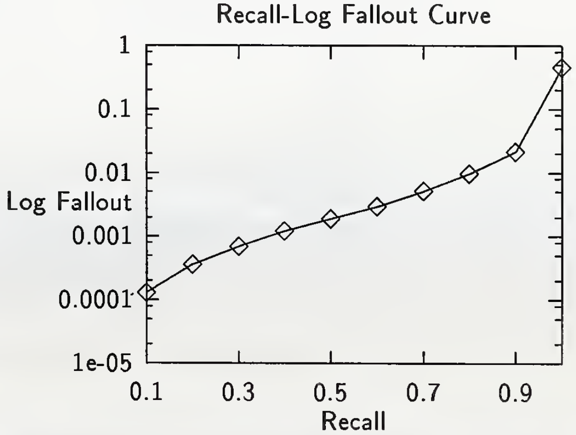
Average precision over all relevant docs	
non-interpolated	0.3983

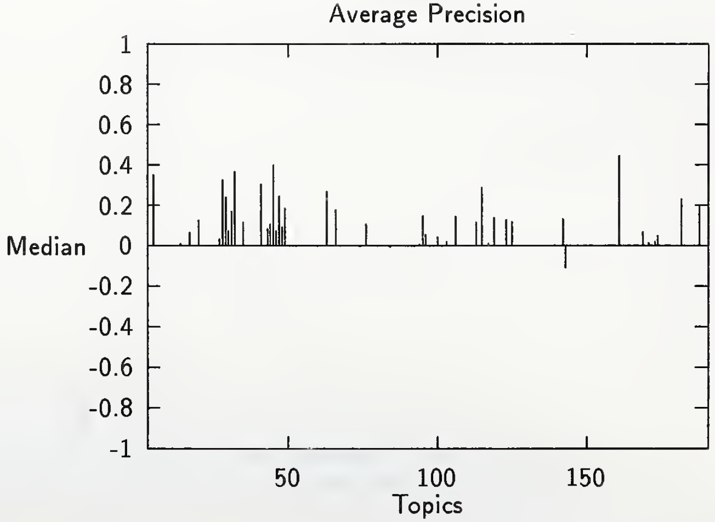
Document Level Averages	
	Precision
At 5 docs	0.6880
At 10 docs	0.6460
At 15 docs	0.6240
At 20 docs	0.6020
At 30 docs	0.5773
At 100 docs	0.4642
At 200 docs	0.3505
At 500 docs	0.1927
At 1000 docs	0.1106

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.4195

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00013
0.20	0.00036
0.30	0.00070
0.40	0.00121
0.50	0.00191
0.60	0.00298
0.70	0.00511
0.80	0.00957
0.90	0.02133
1.00	0.45663







Summary Statistics	
Run Number	itidp1–category A, automatic
Number of Topics	42

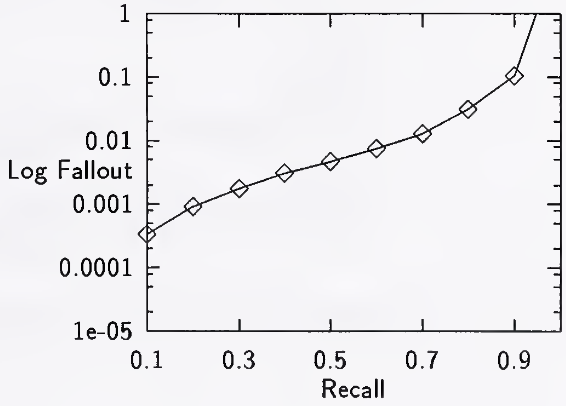
Total number of documents over all topics	
Retrieved:	42000
Relevant:	5091
Rel_ret:	3537

Recall Level Precision Averages	
Recall	Precision
0.00	0.6882
0.10	0.4499
0.20	0.3763
0.30	0.3209
0.40	0.2641
0.50	0.2257
0.60	0.1821
0.70	0.1304
0.80	0.0660
0.90	0.0234
1.00	0.0002
Average precision over all relevant docs	
non-interpolated	0.2282

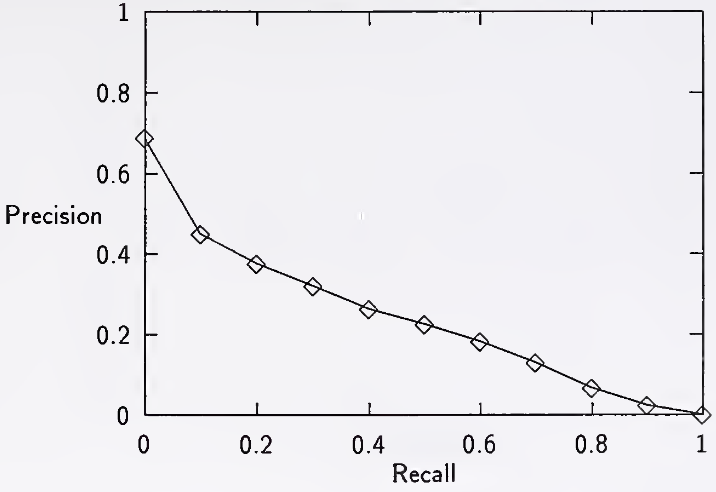
Document Level Averages	
	Precision
At 5 docs	0.5000
At 10 docs	0.4714
At 15 docs	0.4206
At 20 docs	0.4071
At 30 docs	0.3794
At 100 docs	0.2919
At 200 docs	0.2270
At 500 docs	0.1388
At 1000 docs	0.0842
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2782

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00034
0.20	0.00091
0.30	0.00175
0.40	0.00307
0.50	0.00473
0.60	0.00743
0.70	0.01287
0.80	0.03121
0.90	0.10353
1.00	13.77893

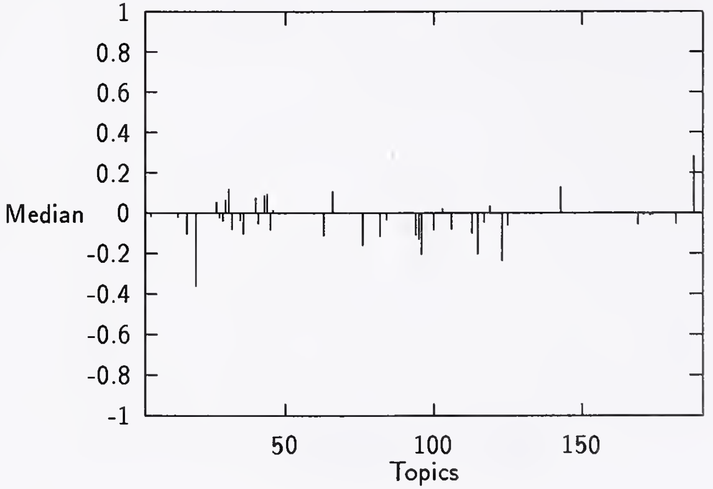
Recall-Log Fallout Curve



Recall-Precision Curve



Average Precision



Summary Statistics	
Run Number	itidp2–category A, automatic
Number of Topics	42

Total number of documents over all topics	
Retrieved:	42000
Relevant:	5091
Rel_ret:	3104

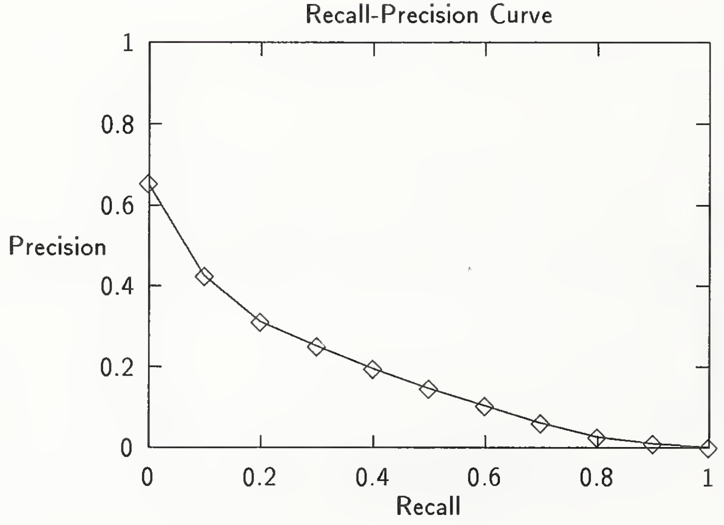
Recall Level Precision Averages	
Recall	Precision
0.00	0.6528
0.10	0.4252
0.20	0.3123
0.30	0.2510
0.40	0.1961
0.50	0.1457
0.60	0.1041
0.70	0.0604
0.80	0.0249
0.90	0.0096
1.00	0.0007

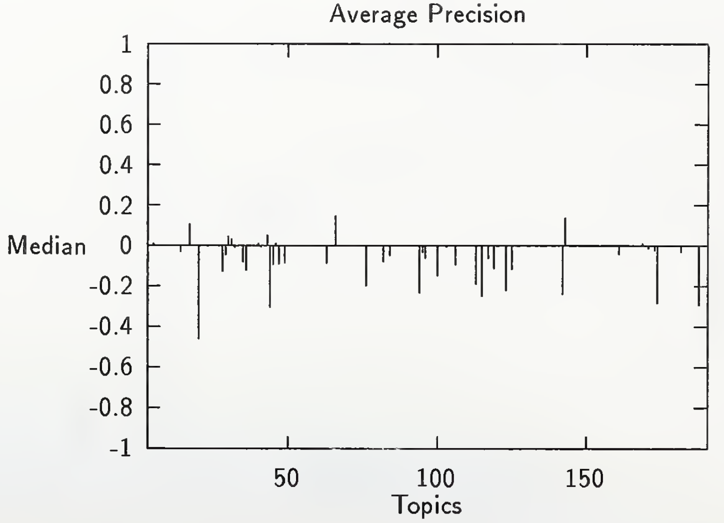
Average precision over all relevant docs	
non-interpolated	0.1768

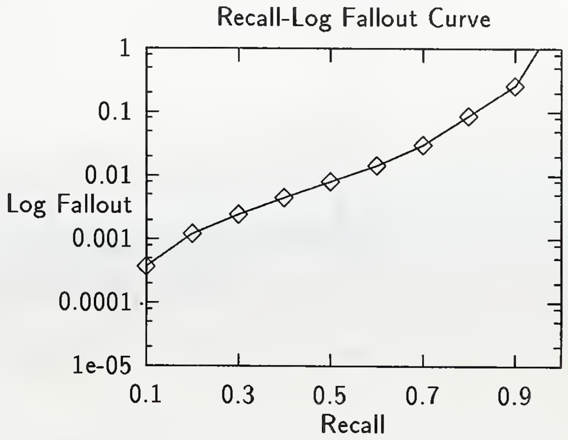
Document Level Averages	
	Precision
At 5 docs	0.4714
At 10 docs	0.4524
At 15 docs	0.4254
At 20 docs	0.4083
At 30 docs	0.3738
At 100 docs	0.2664
At 200 docs	0.1973
At 500 docs	0.1177
At 1000 docs	0.0739

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2221

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00037
0.20	0.00121
0.30	0.00247
0.40	0.00452
0.50	0.00808
0.60	0.01423
0.70	0.03001
0.80	0.08635
0.90	0.25593
1.00	3.93487







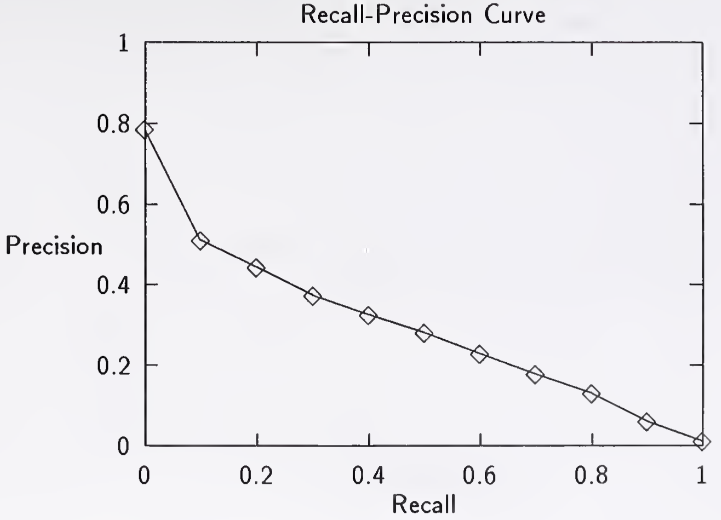
Summary Statistics	
Run Number	losPA2–category A, automatic
Number of Topics	50

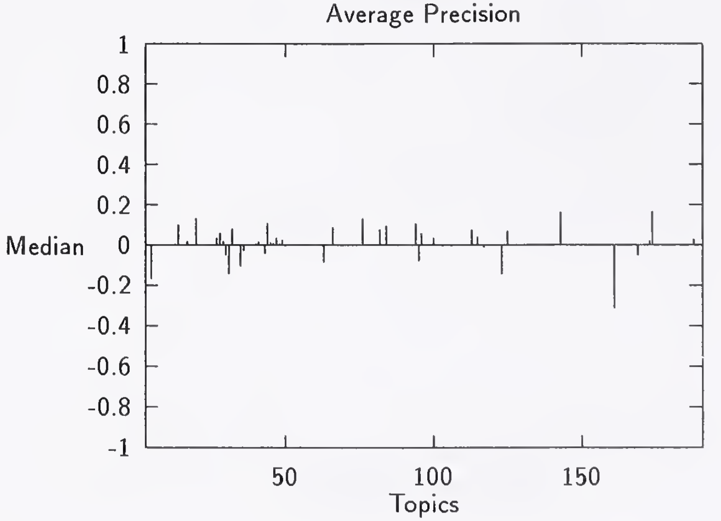
Total number of documents over all topics	
Retrieved:	48708
Relevant:	6576
Rel_ret:	4993

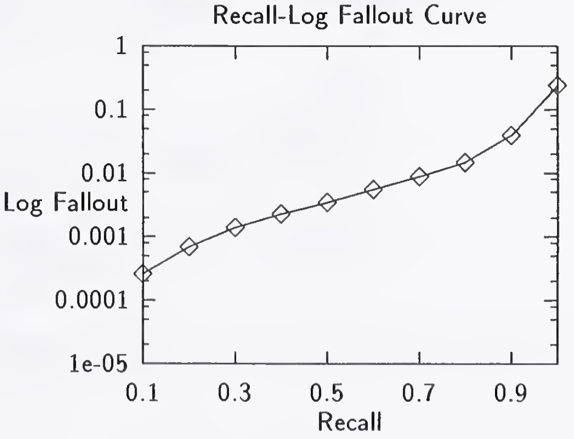
Recall Level Precision Averages	
Recall	Precision
0.00	0.7848
0.10	0.5105
0.20	0.4436
0.30	0.3738
0.40	0.3262
0.50	0.2817
0.60	0.2293
0.70	0.1781
0.80	0.1303
0.90	0.0598
1.00	0.0114
Average precision over all relevant docs	
non-interpolated	0.2793

Document Level Averages	
	Precision
At 5 docs	0.5160
At 10 docs	0.4940
At 15 docs	0.4840
At 20 docs	0.4700
At 30 docs	0.4433
At 100 docs	0.3578
At 200 docs	0.2757
At 500 docs	0.1651
At 1000 docs	0.0999
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3108

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00026
0.20	0.00069
0.30	0.00139
0.40	0.00228
0.50	0.00351
0.60	0.00556
0.70	0.00890
0.80	0.01472
0.90	0.03900
1.00	0.23903





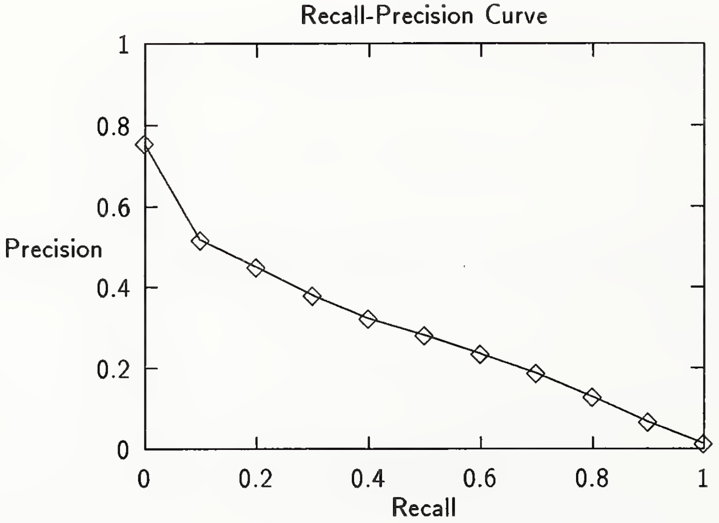


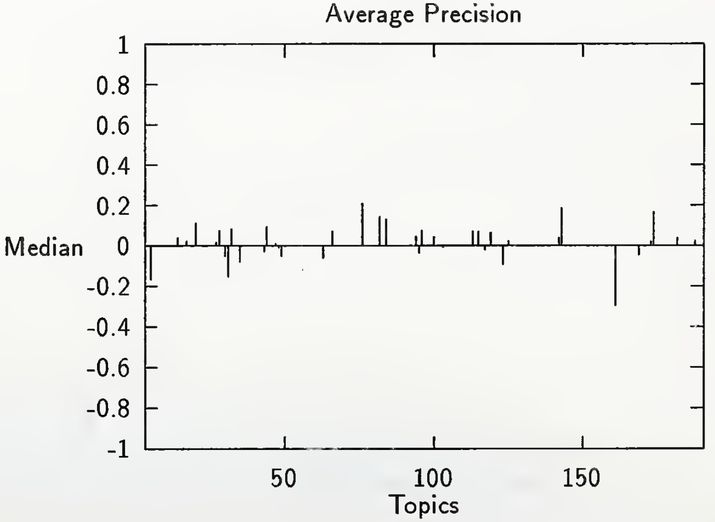
Summary Statistics	
Run Number	losPA3–category A, automatic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	49358
Relevant:	6576
Rel_ret:	4957

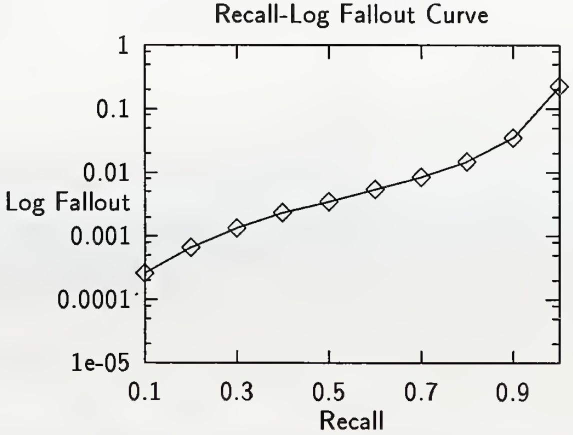
Recall Level Precision Averages	
Recall	Precision
0.00	0.7553
0.10	0.5174
0.20	0.4505
0.30	0.3796
0.40	0.3234
0.50	0.2820
0.60	0.2354
0.70	0.1874
0.80	0.1293
0.90	0.0668
1.00	0.0122
Average precision over all relevant docs	
non-interpolated	0.2834

Document Level Averages	
	Precision
At 5 docs	0.5160
At 10 docs	0.4960
At 15 docs	0.4840
At 20 docs	0.4670
At 30 docs	0.4527
At 100 docs	0.3550
At 200 docs	0.2828
At 500 docs	0.1651
At 1000 docs	0.0991
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3161

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00026
0.20	0.00067
0.30	0.00135
0.40	0.00231
0.50	0.00351
0.60	0.00537
0.70	0.00837
0.80	0.01485
0.90	0.03466
1.00	0.22317





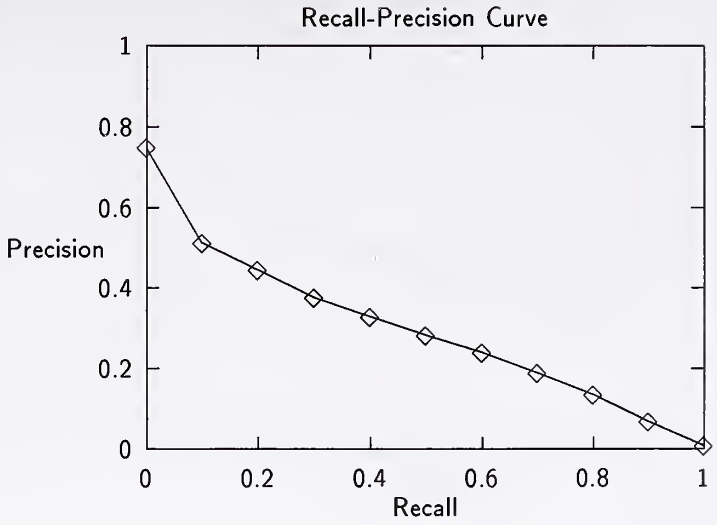


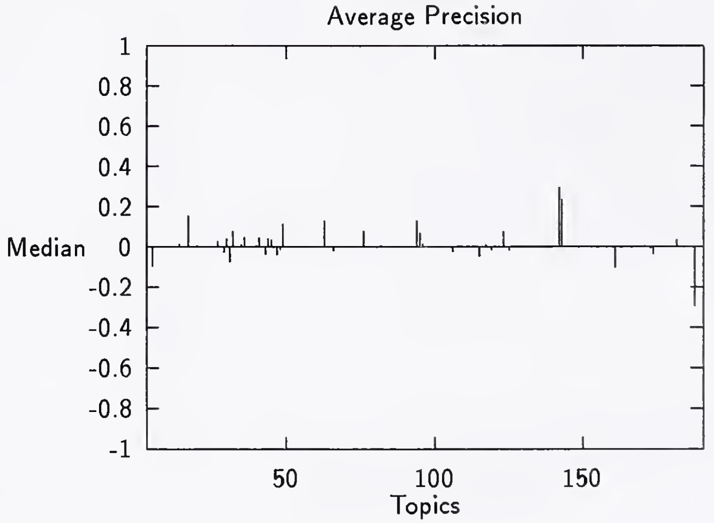
Summary Statistics	
Run Number	nyuge1–category A, automatic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	6576
Rel_ret:	5078

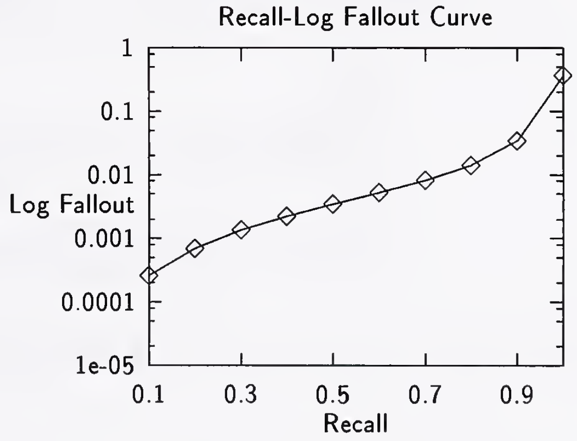
Recall Level Precision Averages	
Recall	Precision
0.00	0.7483
0.10	0.5114
0.20	0.4453
0.30	0.3770
0.40	0.3290
0.50	0.2823
0.60	0.2397
0.70	0.1893
0.80	0.1358
0.90	0.0683
1.00	0.0074
Average precision over all relevant docs	
non-interpolated	0.2838

Document Level Averages	
	Precision
At 5 docs	0.5560
At 10 docs	0.5000
At 15 docs	0.4880
At 20 docs	0.4680
At 30 docs	0.4600
At 100 docs	0.3658
At 200 docs	0.2886
At 500 docs	0.1701
At 1000 docs	0.1016
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3112

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00026
0.20	0.00069
0.30	0.00137
0.40	0.00225
0.50	0.00350
0.60	0.00525
0.70	0.00826
0.80	0.01403
0.90	0.03384
1.00	0.36972







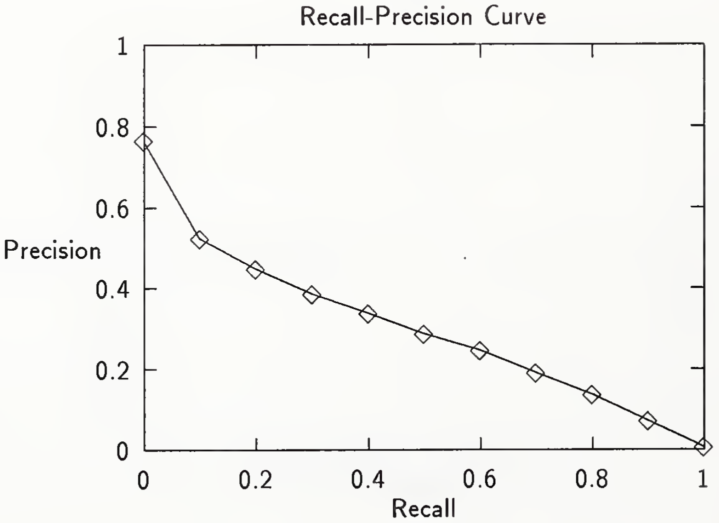
Summary Statistics	
Run Number	nyuge2–category A, automatic
Number of Topics	50

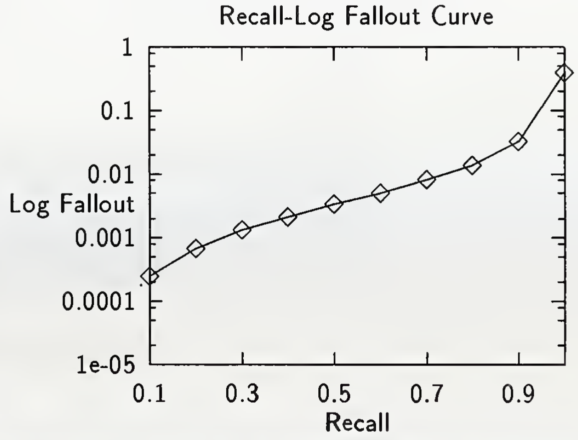
Total number of documents over all topics	
Retrieved:	50000
Relevant:	6576
Rel_ret:	5112

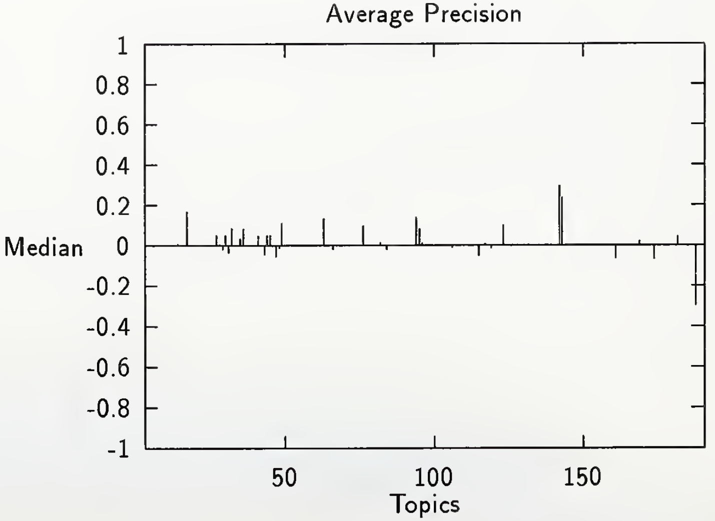
Recall Level Precision Averages	
Recall	Precision
0.00	0.7641
0.10	0.5236
0.20	0.4491
0.30	0.3857
0.40	0.3389
0.50	0.2876
0.60	0.2469
0.70	0.1910
0.80	0.1372
0.90	0.0711
1.00	0.0070
Average precision over all relevant docs	
non-interpolated	0.2913

Document Level Averages	
	Precision
At 5 docs	0.5680
At 10 docs	0.5220
At 15 docs	0.4933
At 20 docs	0.4880
At 30 docs	0.4680
At 100 docs	0.3726
At 200 docs	0.2931
At 500 docs	0.1718
At 1000 docs	0.1022
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3191

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00025
0.20	0.00068
0.30	0.00132
0.40	0.00215
0.50	0.00341
0.60	0.00504
0.70	0.00817
0.80	0.01387
0.90	0.03241
1.00	0.39101





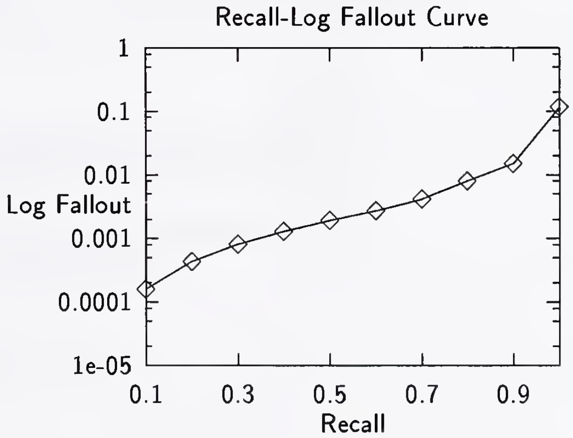


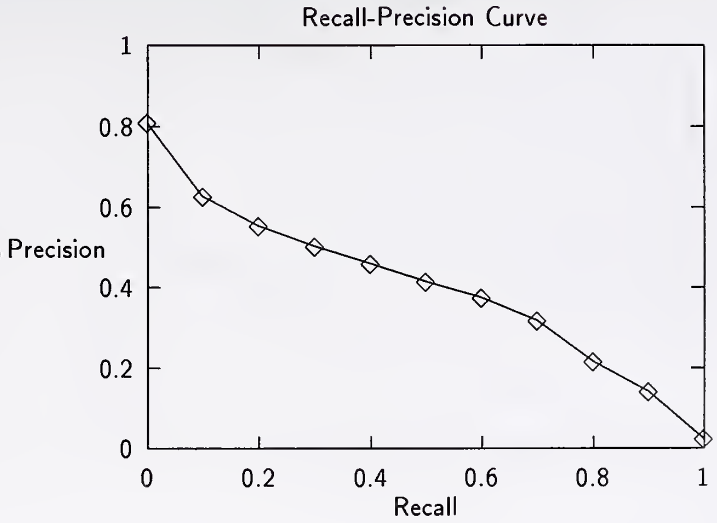
Summary Statistics	
Run Number	pircsC–category A, automatic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	6576
Rel_ret:	5635

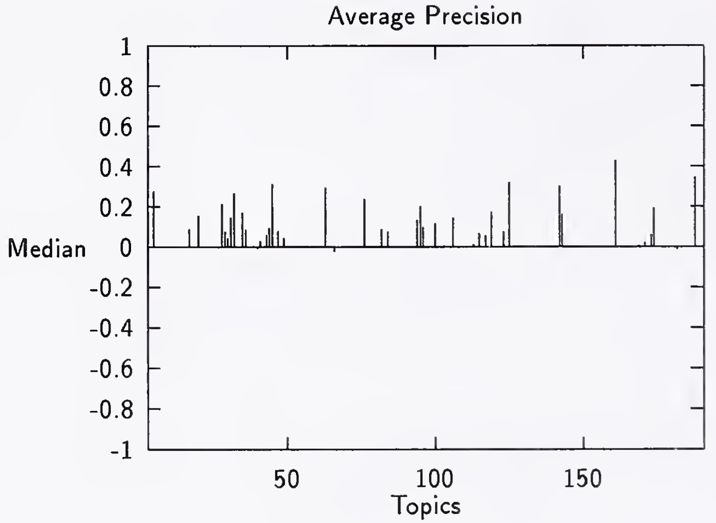
Recall Level Precision Averages	
Recall	Precision
0.00	0.8098
0.10	0.6263
0.20	0.5540
0.30	0.5029
0.40	0.4591
0.50	0.4147
0.60	0.3749
0.70	0.3180
0.80	0.2162
0.90	0.1416
1.00	0.0231
Average precision over all relevant docs	
non-interpolated	0.3909

Document Level Averages	
	Precision
At 5 docs	0.6400
At 10 docs	0.6020
At 15 docs	0.5813
At 20 docs	0.5580
At 30 docs	0.5353
At 100 docs	0.4362
At 200 docs	0.3423
At 500 docs	0.1968
At 1000 docs	0.1127
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.4091

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00016
0.20	0.00044
0.30	0.00082
0.40	0.00130
0.50	0.00195
0.60	0.00276
0.70	0.00414
0.80	0.00799
0.90	0.01504
1.00	0.11657







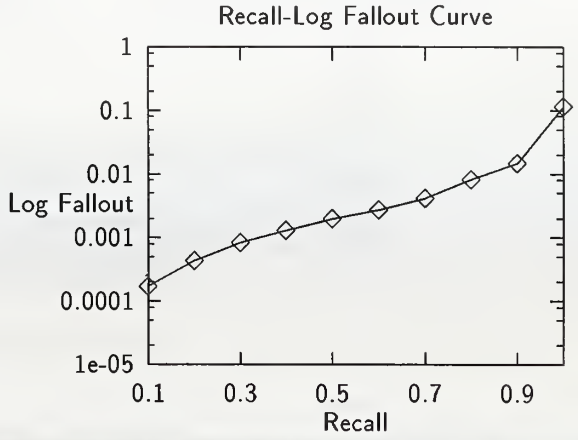
Summary Statistics	
Run Number	pircsL–category A, automatic
Number of Topics	50

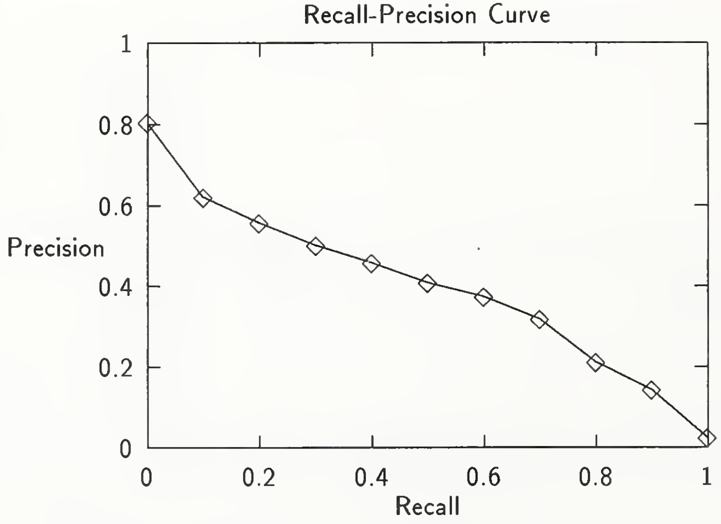
Total number of documents over all topics	
Retrieved:	50000
Relevant:	6576
Rel_ret:	5645

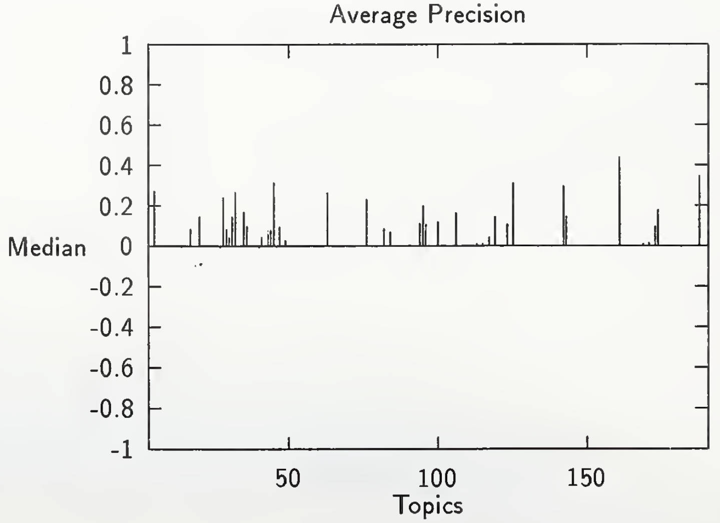
Recall Level Precision Averages	
Recall	Precision
0.00	0.8040
0.10	0.6199
0.20	0.5562
0.30	0.5003
0.40	0.4578
0.50	0.4091
0.60	0.3734
0.70	0.3181
0.80	0.2108
0.90	0.1432
1.00	0.0235
Average precision over all relevant docs	
non-interpolated	0.3901

Document Level Averages	
	Precision
At 5 docs	0.6520
At 10 docs	0.6080
At 15 docs	0.5827
At 20 docs	0.5630
At 30 docs	0.5387
At 100 docs	0.4386
At 200 docs	0.3439
At 500 docs	0.1978
At 1000 docs	0.1129
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.4108

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00017
0.20	0.00044
0.30	0.00083
0.40	0.00131
0.50	0.00199
0.60	0.00278
0.70	0.00414
0.80	0.00826
0.90	0.01484
1.00	0.11453







Summary Statistics	
Run Number	UCF100–category A, manual
Number of Topics	50

Total number of documents over all topics	
Retrieved:	50000
Relevant:	6576
Rel_ret:	4240

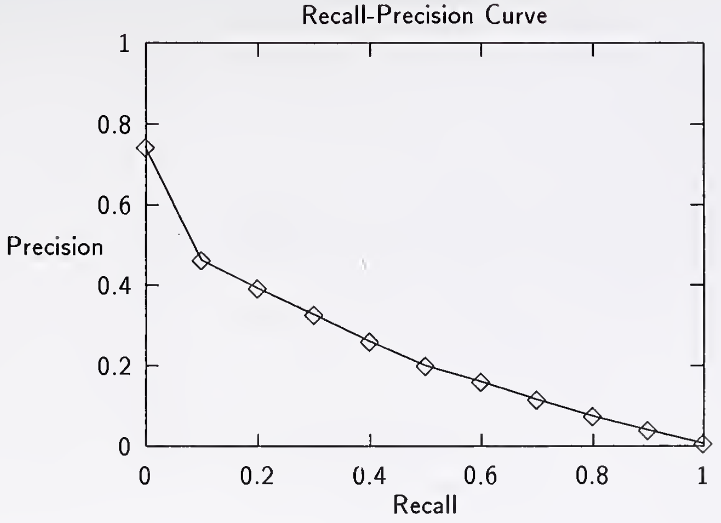
Recall Level Precision Averages	
Recall	Precision
0.00	0.7427
0.10	0.4626
0.20	0.3921
0.30	0.3268
0.40	0.2611
0.50	0.2008
0.60	0.1608
0.70	0.1168
0.80	0.0754
0.90	0.0405
1.00	0.0060

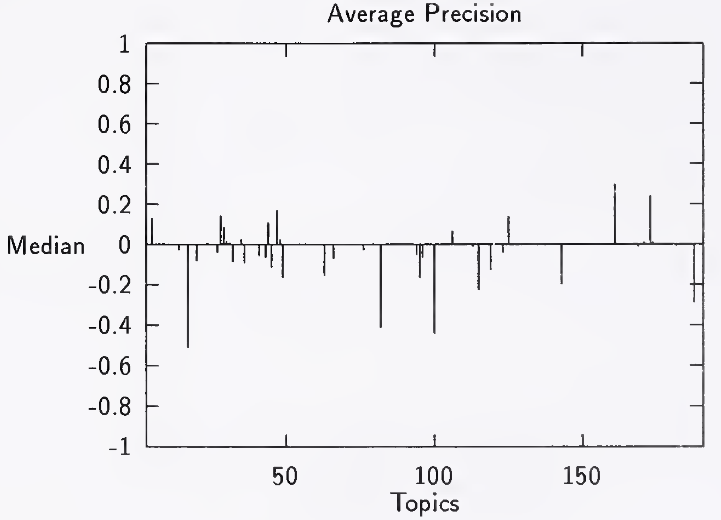
Average precision over all relevant docs	
non-interpolated	0.2285

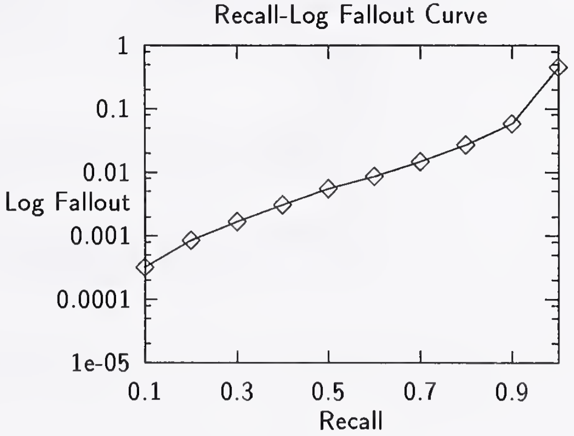
Document Level Averages	
	Precision
At 5 docs	0.5320
At 10 docs	0.5000
At 15 docs	0.4747
At 20 docs	0.4580
At 30 docs	0.4353
At 100 docs	0.3420
At 200 docs	0.2426
At 500 docs	0.1412
At 1000 docs	0.0848

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2695

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00032
0.20	0.00085
0.30	0.00170
0.40	0.00312
0.50	0.00549
0.60	0.00863
0.70	0.01459
0.80	0.02704
0.90	0.05877
1.00	0.45663







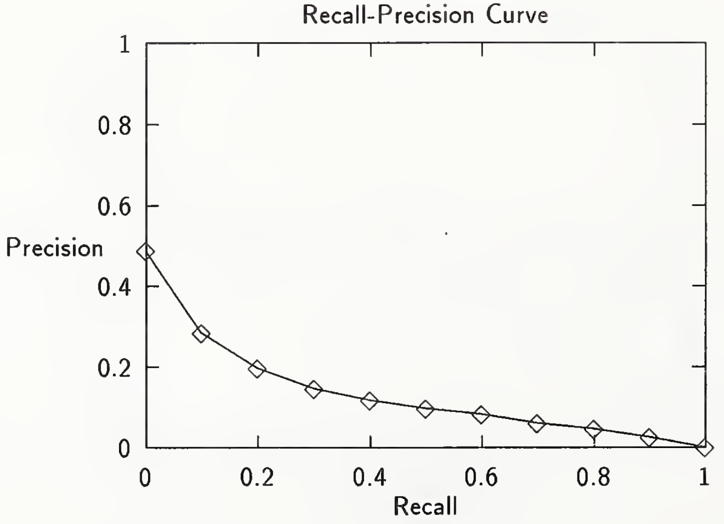
Summary Statistics	
Run Number	uwgcl1–category A, manual
Number of Topics	50

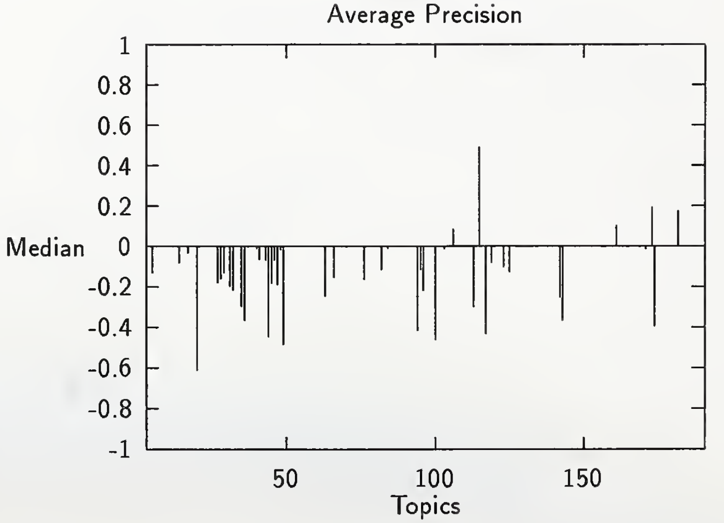
Total number of documents over all topics	
Retrieved:	22769
Relevant:	6576
Rel_ret:	1658

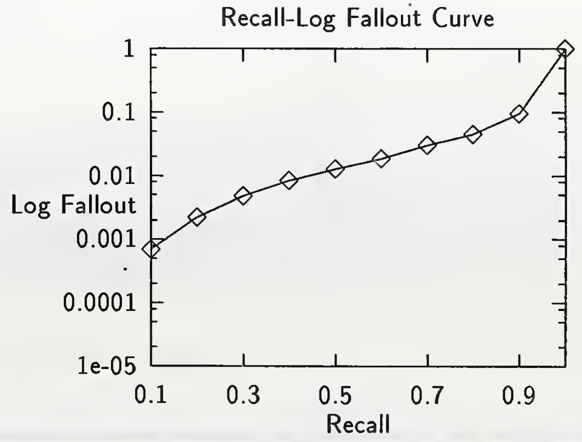
Recall Level Precision Averages	
Recall	Precision
0.00	0.4883
0.10	0.2849
0.20	0.1979
0.30	0.1463
0.40	0.1170
0.50	0.0976
0.60	0.0823
0.70	0.0602
0.80	0.0463
0.90	0.0252
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.1188

Document Level Averages	
	Precision
At 5 docs	0.3480
At 10 docs	0.3080
At 15 docs	0.2920
At 20 docs	0.2680
At 30 docs	0.2387
At 100 docs	0.1710
At 200 docs	0.1139
At 500 docs	0.0613
At 1000 docs	0.0332
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1649

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00069
0.20	0.00223
0.30	0.00483
0.40	0.00832
0.50	0.01274
0.60	0.01844
0.70	0.03012
0.80	0.04542
0.90	0.09596
1.00	1.00000







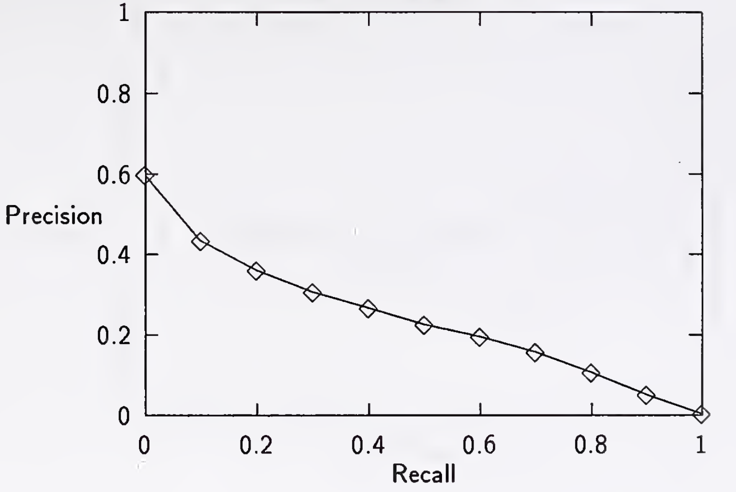
Summary Statistics	
Run Number	virtu3–category A, automatic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	6576
Rel_ret:	4265

Recall Level Precision Averages	
Recall	Precision
0.00	0.5983
0.10	0.4344
0.20	0.3609
0.30	0.3069
0.40	0.2670
0.50	0.2263
0.60	0.1959
0.70	0.1579
0.80	0.1060
0.90	0.0514
1.00	0.0031
Average precision over all relevant docs	
non-interpolated	0.2270

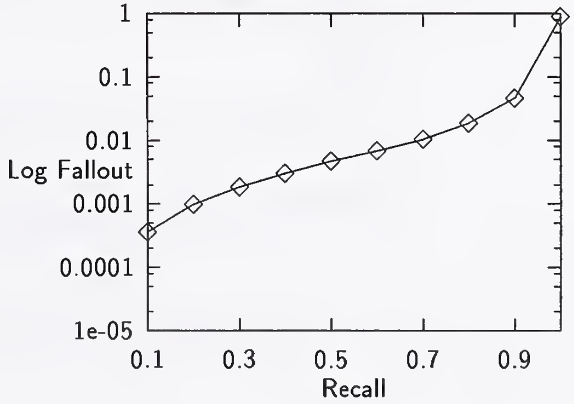
Document Level Averages	
	Precision
At 5 docs	0.3600
At 10 docs	0.3660
At 15 docs	0.3707
At 20 docs	0.3680
At 30 docs	0.3560
At 100 docs	0.2988
At 200 docs	0.2282
At 500 docs	0.1398
At 1000 docs	0.0853
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2637

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00036
0.20	0.00098
0.30	0.00187
0.40	0.00303
0.50	0.00471
0.60	0.00679
0.70	0.01029
0.80	0.01860
0.90	0.04578
1.00	0.88639

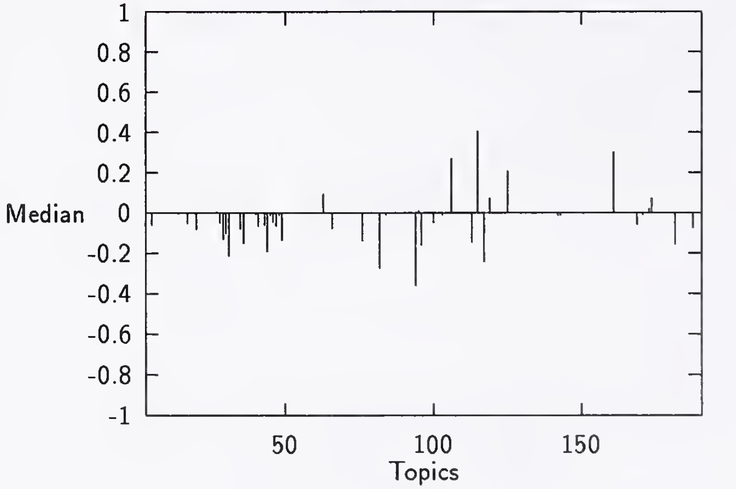
Recall-Precision Curve



Recall-Log Fallout Curve



Average Precision



Summary Statistics	
Run Number	xerox1–category A, automatic
Number of Topics	50

Total number of documents over all topics	
Retrieved:	50000
Relevant:	6576
Rel_ret:	5633

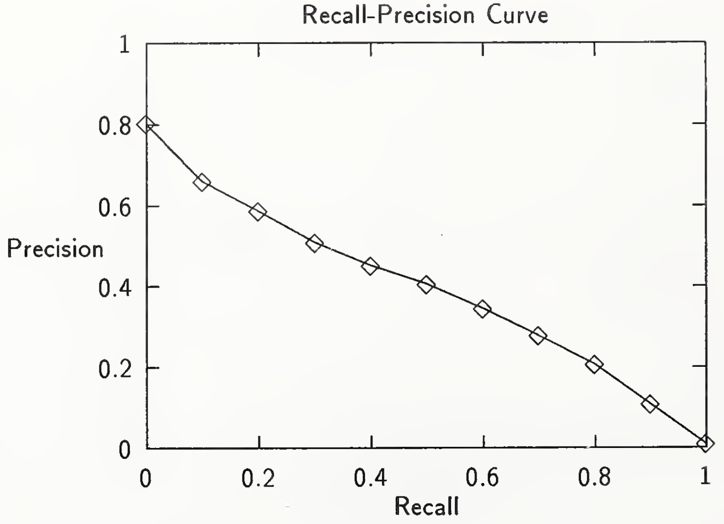
Recall Level Precision Averages	
Recall	Precision
0.00	0.8023
0.10	0.6600
0.20	0.5864
0.30	0.5089
0.40	0.4522
0.50	0.4062
0.60	0.3455
0.70	0.2785
0.80	0.2064
0.90	0.1078
1.00	0.0101

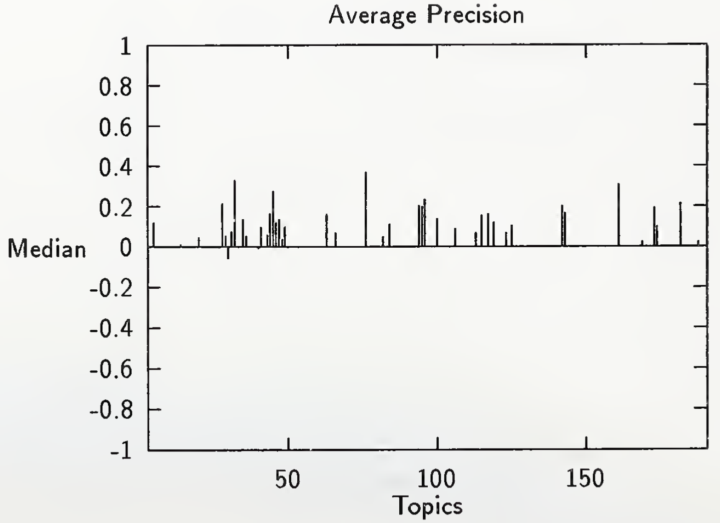
Average precision over all relevant docs	
non-interpolated	0.3841

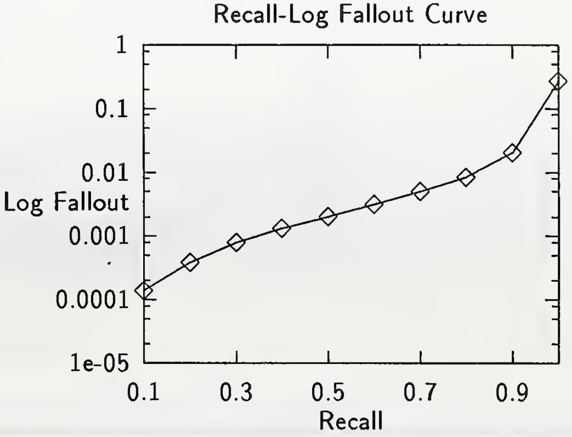
Document Level Averages	
	Precision
At 5 docs	0.6440
At 10 docs	0.6120
At 15 docs	0.5773
At 20 docs	0.5620
At 30 docs	0.5513
At 100 docs	0.4408
At 200 docs	0.3388
At 500 docs	0.1940
At 1000 docs	0.1127

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.3993

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00014
0.20	0.00039
0.30	0.00080
0.40	0.00134
0.50	0.00201
0.60	0.00313
0.70	0.00500
0.80	0.00848
0.90	0.02053
1.00	0.27015





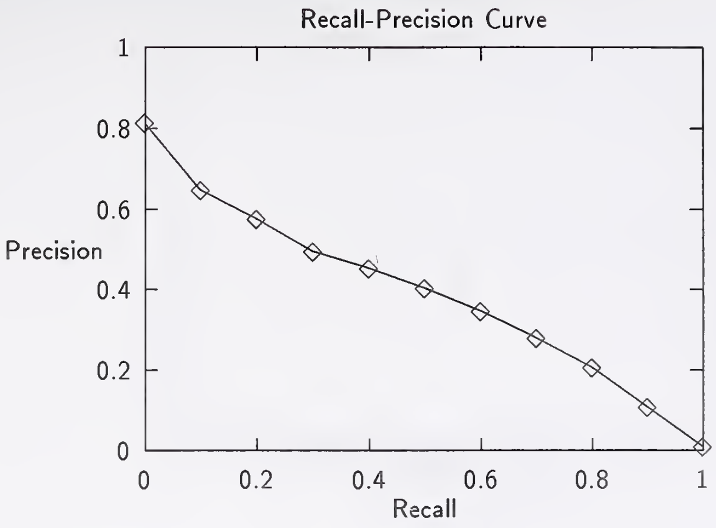


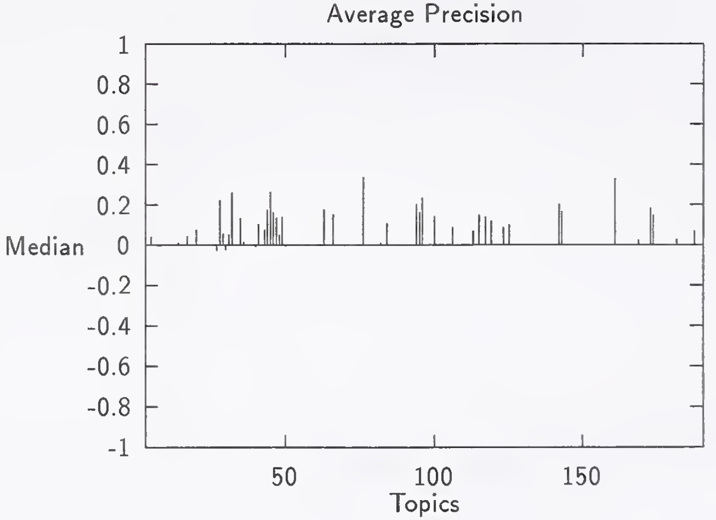
Summary Statistics	
Run Number	xerox2–category A, automatic
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	6576
Rel_ret:	5633

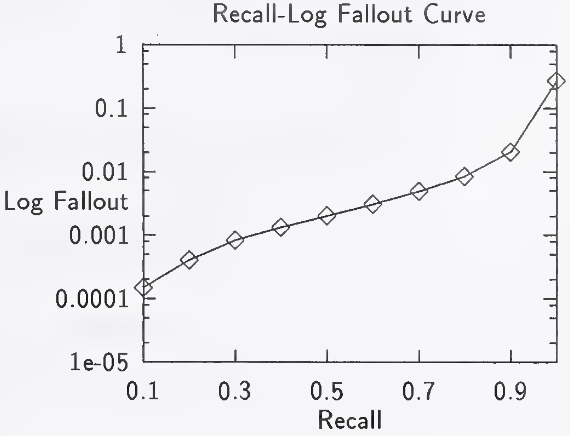
Recall Level Precision Averages	
Recall	Precision
0.00	0.8147
0.10	0.6468
0.20	0.5750
0.30	0.4947
0.40	0.4529
0.50	0.4032
0.60	0.3472
0.70	0.2797
0.80	0.2071
0.90	0.1076
1.00	0.0101
Average precision over all relevant docs	
non-interpolated	0.3811

Document Level Averages	
	Precision
At 5 docs	0.6080
At 10 docs	0.5900
At 15 docs	0.5720
At 20 docs	0.5640
At 30 docs	0.5467
At 100 docs	0.4366
At 200 docs	0.3391
At 500 docs	0.1941
At 1000 docs	0.1127
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.4039

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00015
0.20	0.00041
0.30	0.00084
0.40	0.00133
0.50	0.00204
0.60	0.00311
0.70	0.00497
0.80	0.00844
0.90	0.02057
1.00	0.27015







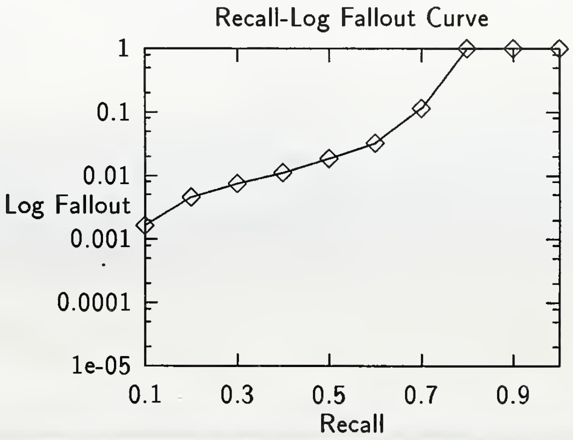
Summary Statistics	
Run Number	ORAdL1–category B, automatic
Number of Topics	50

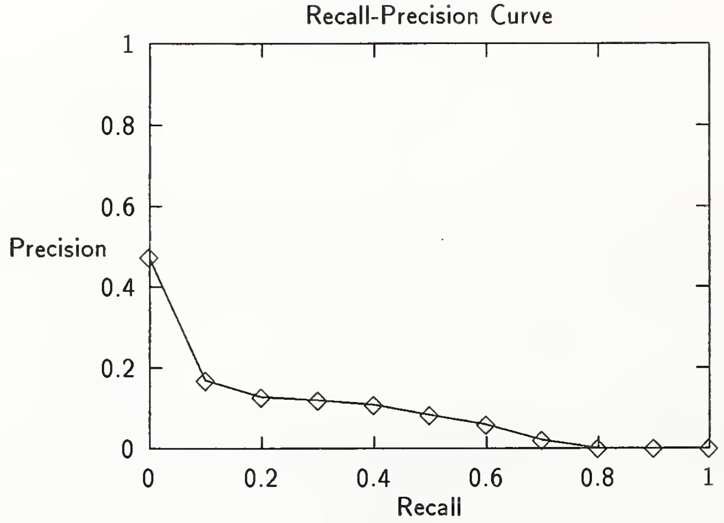
Total number of documents over all topics	
Retrieved:	9951
Relevant:	6099
Rel_ret:	1057

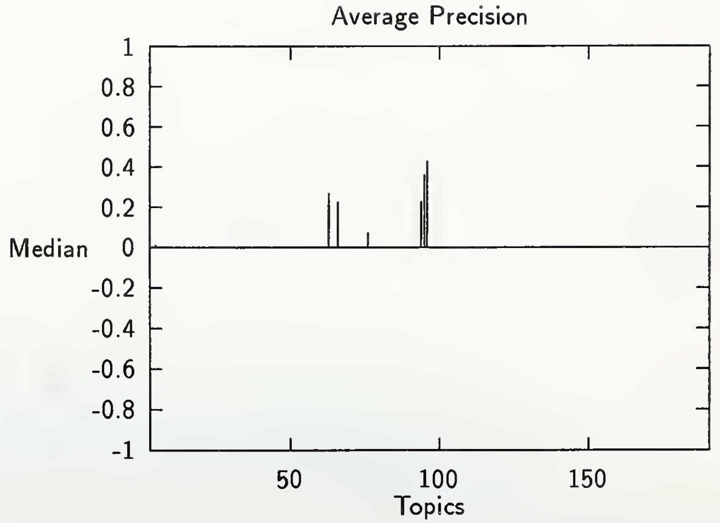
Recall Level Precision Averages	
Recall	Precision
0.00	0.4730
0.10	0.1690
0.20	0.1278
0.30	0.1193
0.40	0.1084
0.50	0.0825
0.60	0.0587
0.70	0.0203
0.80	0.0000
0.90	0.0000
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.0812

Document Level Averages	
	Precision
At 5 docs	0.2920
At 10 docs	0.2420
At 15 docs	0.2147
At 20 docs	0.1950
At 30 docs	0.1787
At 100 docs	0.1374
At 200 docs	0.0961
At 500 docs	0.0410
At 1000 docs	0.0211
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1098

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00165
0.20	0.00458
0.30	0.00743
0.40	0.01103
0.50	0.01865
0.60	0.03227
0.70	0.11329
0.80	1.00000
0.90	1.00000
1.00	1.00000







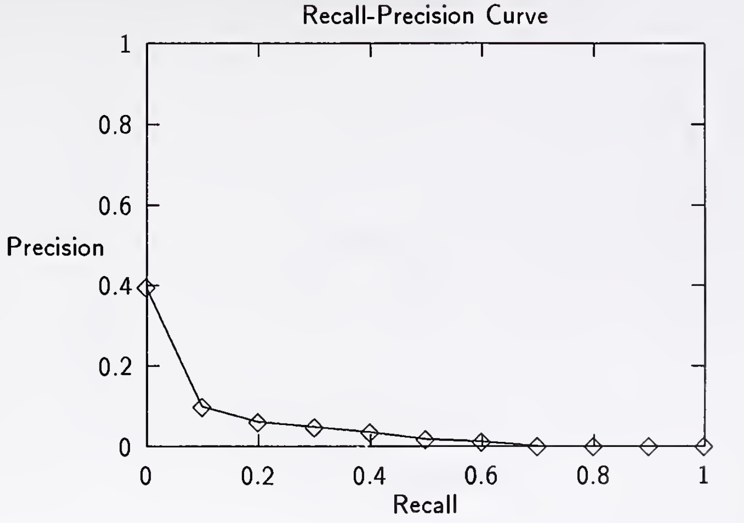
Summary Statistics	
Run Number	ORAdL2–category B, automatic
Number of Topics	50

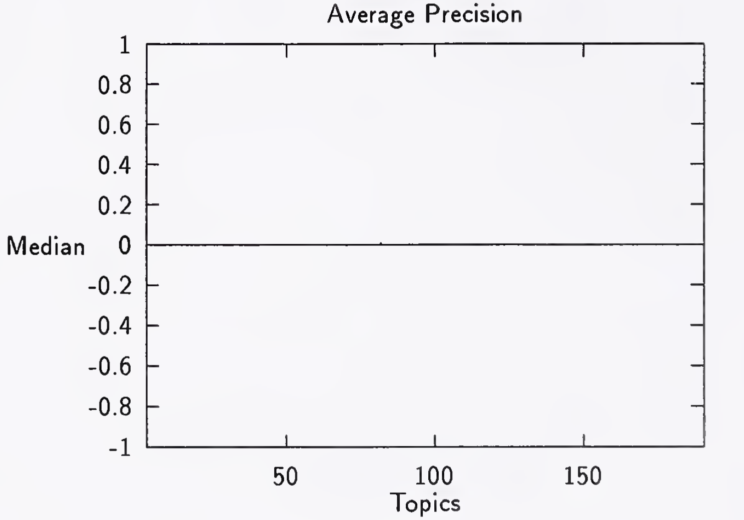
Total number of documents over all topics	
Retrieved:	29893
Relevant:	6099
Rel_ret:	1346

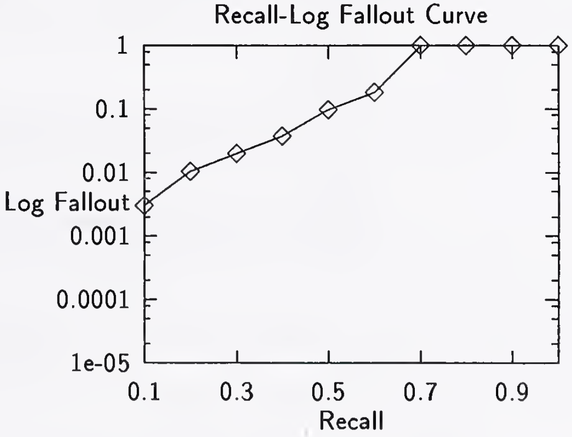
Recall Level Precision Averages	
Recall	Precision
0.00	0.3963
0.10	0.0990
0.20	0.0610
0.30	0.0478
0.40	0.0343
0.50	0.0169
0.60	0.0109
0.70	0.0000
0.80	0.0000
0.90	0.0000
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.0355

Document Level Averages	
	Precision
At 5 docs	0.1480
At 10 docs	0.1400
At 15 docs	0.1267
At 20 docs	0.1110
At 30 docs	0.1047
At 100 docs	0.0750
At 200 docs	0.0631
At 500 docs	0.0412
At 1000 docs	0.0269
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0741

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00305
0.20	0.01032
0.30	0.02004
0.40	0.03777
0.50	0.09754
0.60	0.18258
0.70	1.00000
0.80	1.00000
0.90	1.00000
1.00	1.00000







Summary Statistics	
Run Number	ORAdL1–category B, automatic
Number of Topics	50

Total number of documents over all topics	
Retrieved:	9951
Relevant:	6099
Rel_ret:	1057

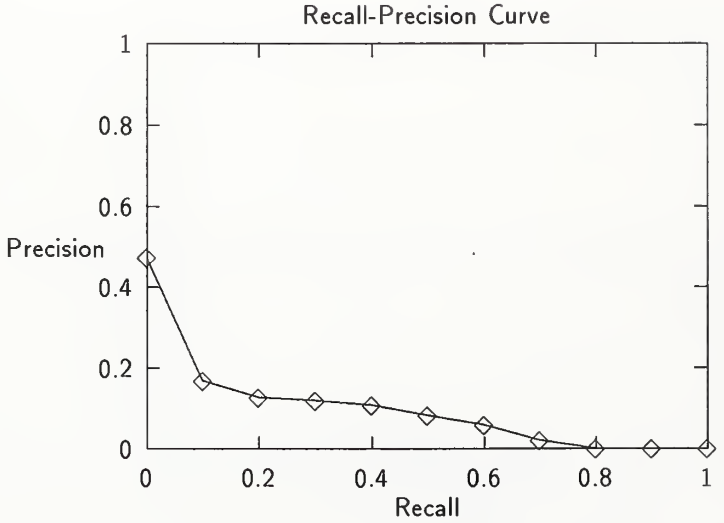
Recall Level Precision Averages	
Recall	Precision
0.00	0.4730
0.10	0.1690
0.20	0.1278
0.30	0.1193
0.40	0.1084
0.50	0.0825
0.60	0.0587
0.70	0.0203
0.80	0.0000
0.90	0.0000
1.00	0.0000

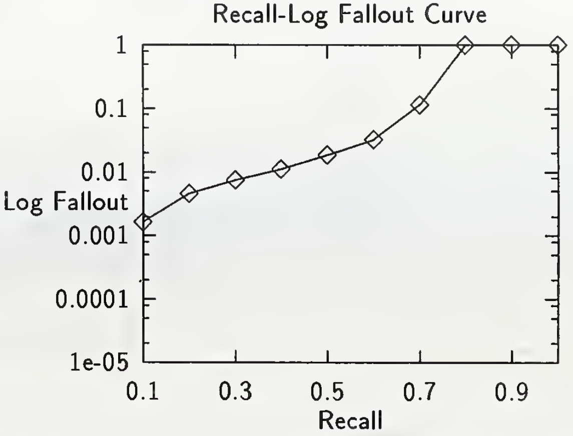
Average precision over all relevant docs	
non-interpolated	0.0812

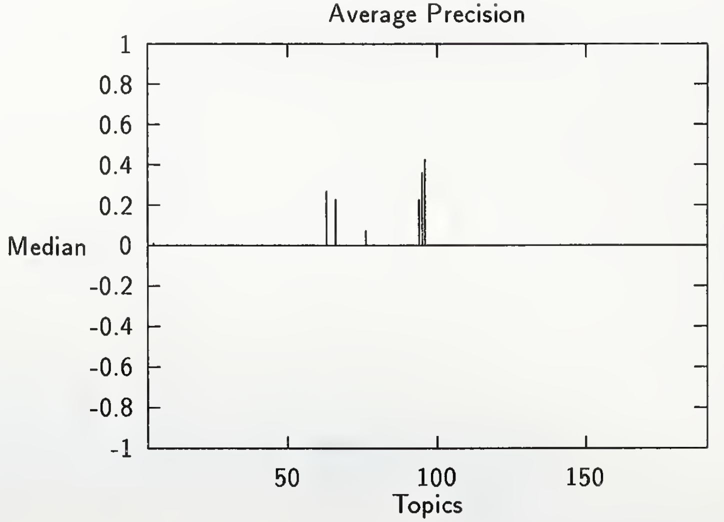
Document Level Averages	
	Precision
At 5 docs	0.2920
At 10 docs	0.2420
At 15 docs	0.2147
At 20 docs	0.1950
At 30 docs	0.1787
At 100 docs	0.1374
At 200 docs	0.0961
At 500 docs	0.0410
At 1000 docs	0.0211

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1098

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00165
0.20	0.00458
0.30	0.00743
0.40	0.01103
0.50	0.01865
0.60	0.03227
0.70	0.11329
0.80	1.00000
0.90	1.00000
1.00	1.00000







Summary Statistics	
Run Number	ORAdL2–category B, automatic
Number of Topics	50

Total number of documents over all topics	
Retrieved:	29893
Relevant:	6099
Rel_ret:	1346

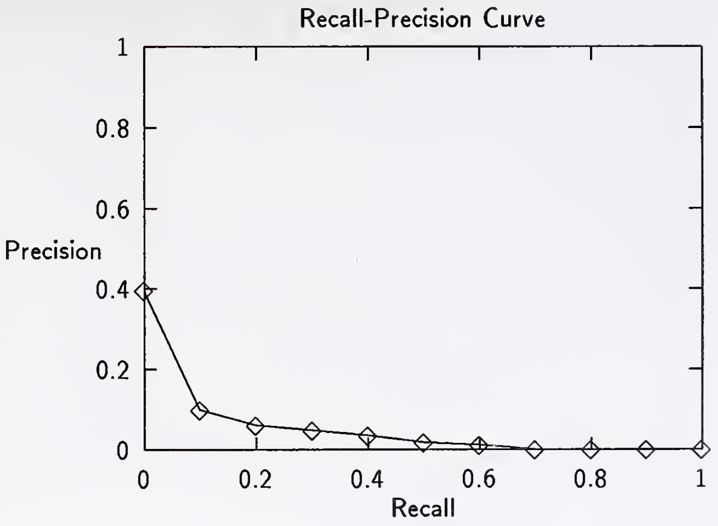
Recall Level Precision Averages	
Recall	Precision
0.00	0.3963
0.10	0.0990
0.20	0.0610
0.30	0.0478
0.40	0.0343
0.50	0.0169
0.60	0.0109
0.70	0.0000
0.80	0.0000
0.90	0.0000
1.00	0.0000

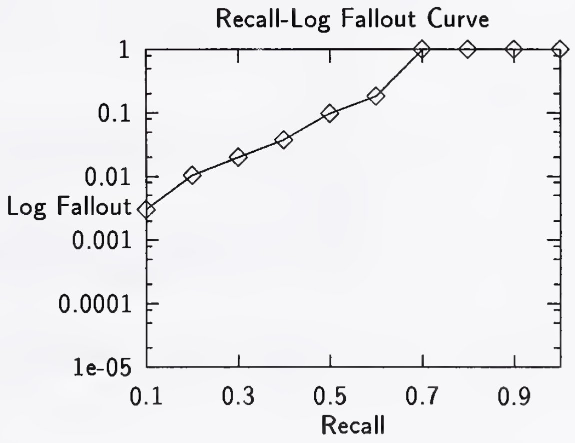
Average precision over all relevant docs	
non-interpolated	0.0355

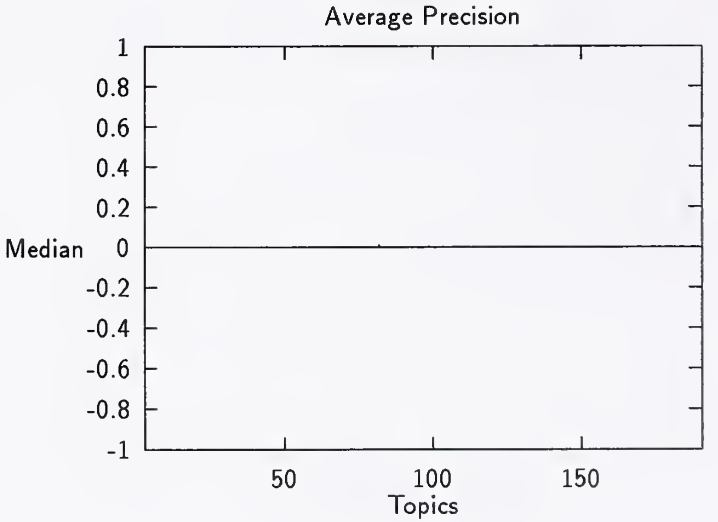
Document Level Averages	
	Precision
At 5 docs	0.1480
At 10 docs	0.1400
At 15 docs	0.1267
At 20 docs	0.1110
At 30 docs	0.1047
At 100 docs	0.0750
At 200 docs	0.0631
At 500 docs	0.0412
At 1000 docs	0.0269

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0741

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00305
0.20	0.01032
0.30	0.02004
0.40	0.03777
0.50	0.09754
0.60	0.18258
0.70	1.00000
0.80	1.00000
0.90	1.00000
1.00	1.00000







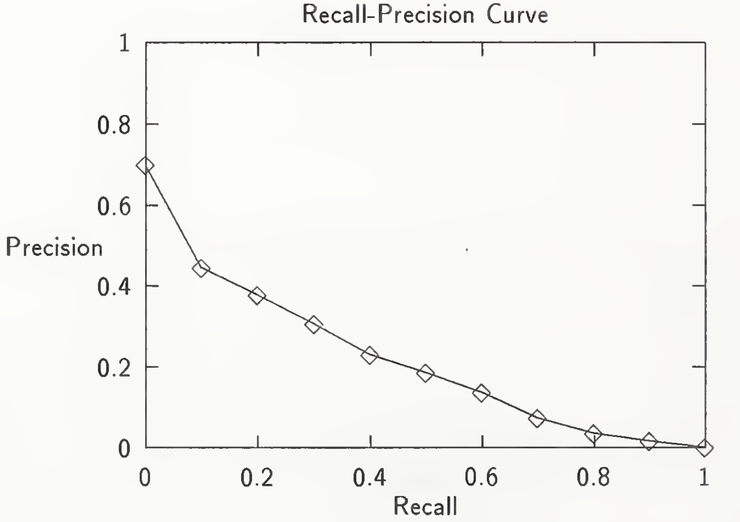
Summary Statistics	
Run Number	inq205–category A, automatic
Number of Topics	49

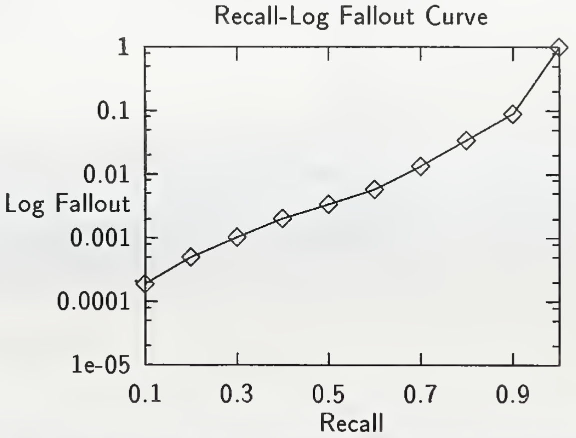
Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	3122

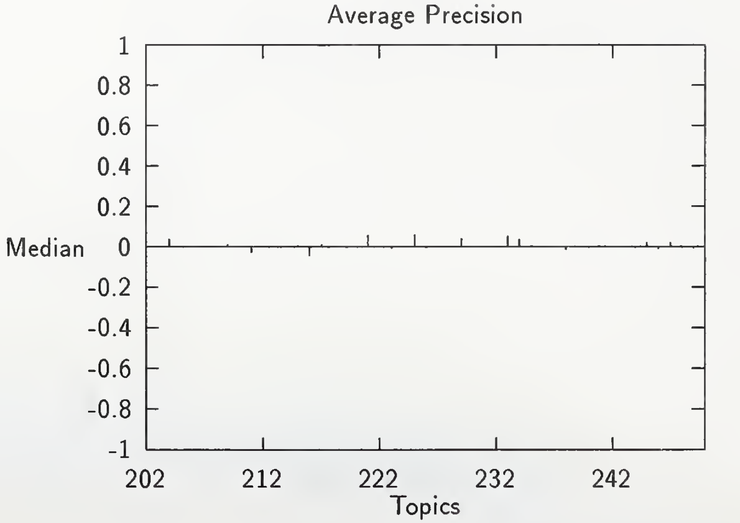
Recall Level Precision Averages	
Recall	Precision
0.00	0.6990
0.10	0.4455
0.20	0.3790
0.30	0.3068
0.40	0.2312
0.50	0.1855
0.60	0.1362
0.70	0.0739
0.80	0.0348
0.90	0.0154
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.2054

Document Level Averages	
	Precision
At 5 docs	0.4980
At 10 docs	0.4367
At 15 docs	0.3959
At 20 docs	0.3786
At 30 docs	0.3456
At 100 docs	0.2357
At 200 docs	0.1789
At 500 docs	0.1042
At 1000 docs	0.0637
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2611

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00019
0.20	0.00050
0.30	0.00104
0.40	0.00204
0.50	0.00336
0.60	0.00583
0.70	0.01343
0.80	0.03398
0.90	0.08811
1.00	1.00000







Summary Statistics	
Run Number	inq206–category A, automatic
Number of Topics	49

Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	3105

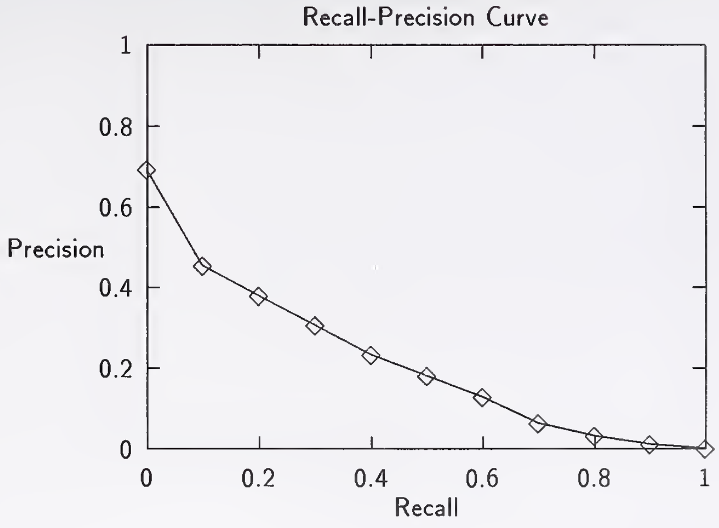
Recall Level Precision Averages	
Recall	Precision
0.00	0.6932
0.10	0.4543
0.20	0.3794
0.30	0.3061
0.40	0.2341
0.50	0.1817
0.60	0.1293
0.70	0.0640
0.80	0.0314
0.90	0.0116
1.00	0.0000

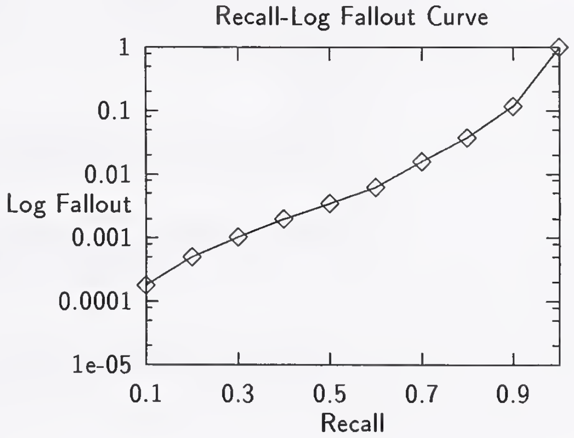
Average precision over all relevant docs	
non-interpolated	0.2047

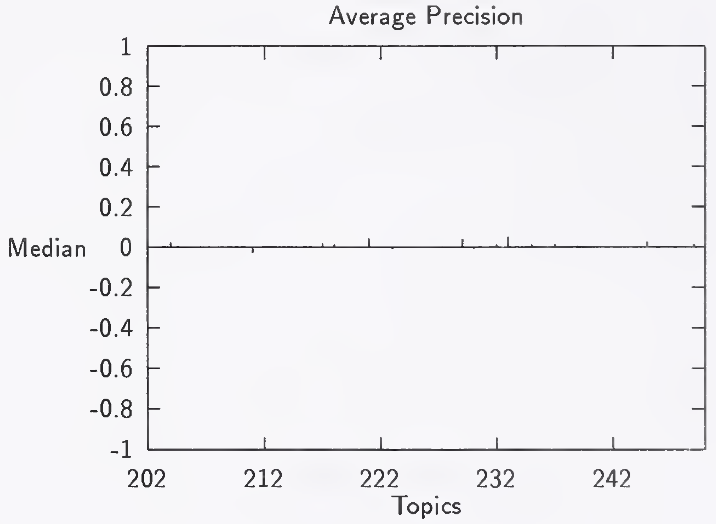
Document Level Averages	
	Precision
At 5 docs	0.5061
At 10 docs	0.4429
At 15 docs	0.4000
At 20 docs	0.3888
At 30 docs	0.3483
At 100 docs	0.2361
At 200 docs	0.1785
At 500 docs	0.1043
At 1000 docs	0.0634

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2603

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00018
0.20	0.00050
0.30	0.00104
0.40	0.00200
0.50	0.00345
0.60	0.00619
0.70	0.01568
0.80	0.03779
0.90	0.11742
1.00	1.00000







Summary Statistics	
Run Number	inq207–category A, automatic
Number of Topics	49

Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	3011

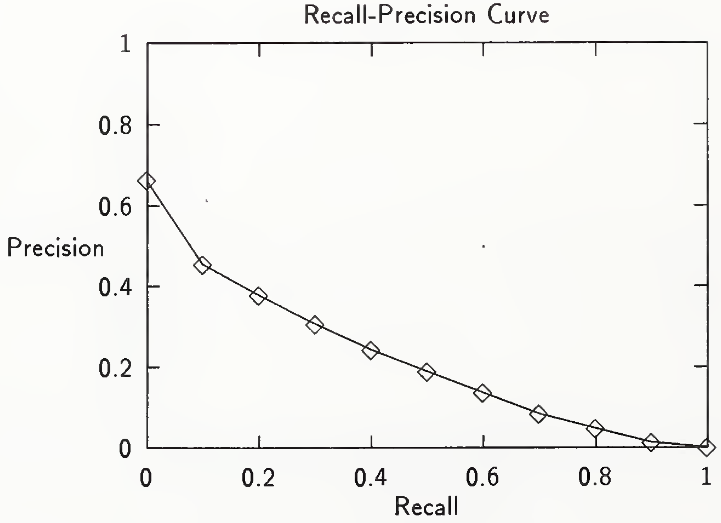
Recall Level Precision Averages	
Recall	Precision
0.00	0.6634
0.10	0.4539
0.20	0.3786
0.30	0.3070
0.40	0.2434
0.50	0.1895
0.60	0.1368
0.70	0.0848
0.80	0.0469
0.90	0.0130
1.00	0.0000

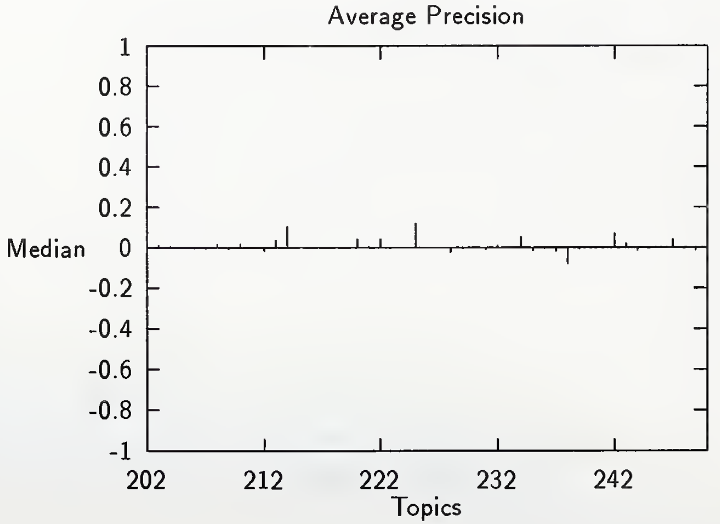
Average precision over all relevant docs	
non-interpolated	0.2085

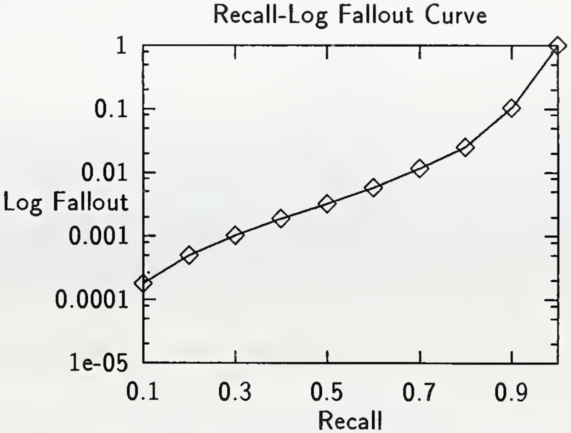
Document Level Averages	
	Precision
At 5 docs	0.4857
At 10 docs	0.4347
At 15 docs	0.4014
At 20 docs	0.3796
At 30 docs	0.3456
At 100 docs	0.2418
At 200 docs	0.1818
At 500 docs	0.1029
At 1000 docs	0.0614

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2589

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00018
0.20	0.00050
0.30	0.00104
0.40	0.00190
0.50	0.00327
0.60	0.00580
0.70	0.01157
0.80	0.02489
0.90	0.10463
1.00	1.00000





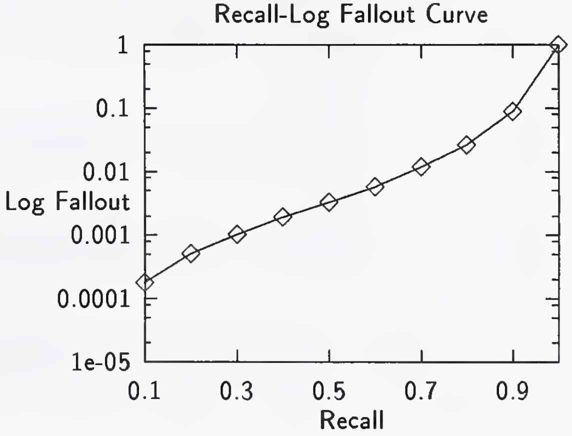


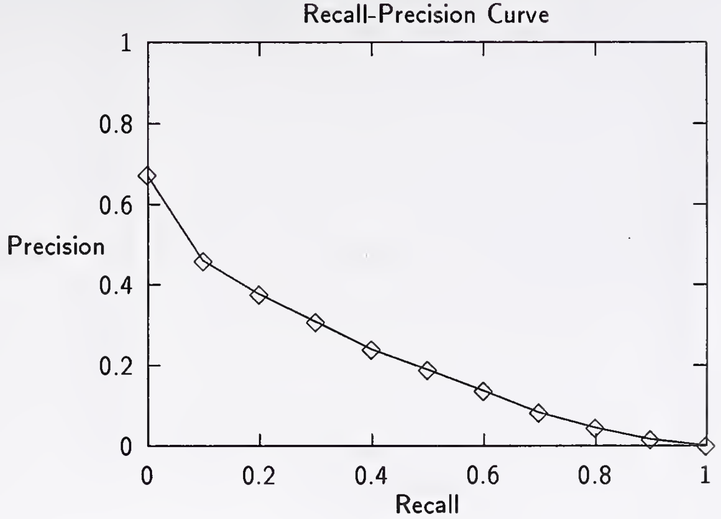
Summary Statistics	
Run Number	inq208–category A, automatic
Number of Topics	49
Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	3033

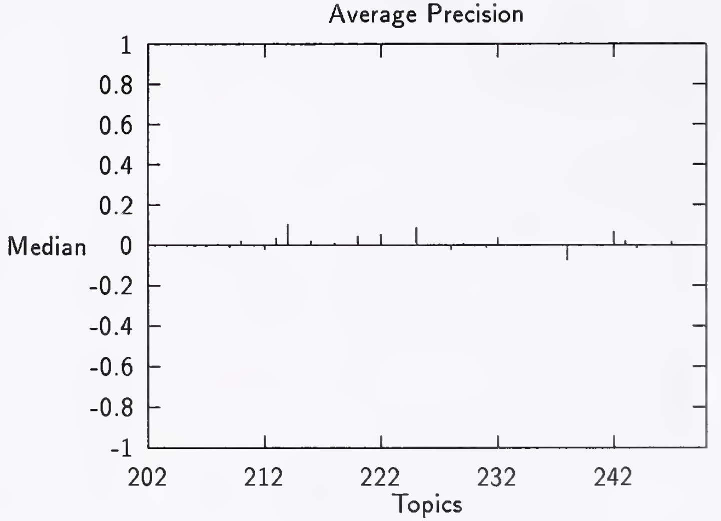
Recall Level Precision Averages	
Recall	Precision
0.00	0.6724
0.10	0.4590
0.20	0.3761
0.30	0.3089
0.40	0.2398
0.50	0.1884
0.60	0.1368
0.70	0.0830
0.80	0.0441
0.90	0.0154
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.2084

Document Level Averages	
	Precision
At 5 docs	0.4776
At 10 docs	0.4327
At 15 docs	0.3973
At 20 docs	0.3898
At 30 docs	0.3503
At 100 docs	0.2427
At 200 docs	0.1805
At 500 docs	0.1029
At 1000 docs	0.0619
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2620

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00018
0.20	0.00051
0.30	0.00103
0.40	0.00194
0.50	0.00330
0.60	0.00580
0.70	0.01184
0.80	0.02655
0.90	0.08811
1.00	1.00000







Summary Statistics	
Run Number	inq209–category A, automatic
Number of Topics	49

Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	3105

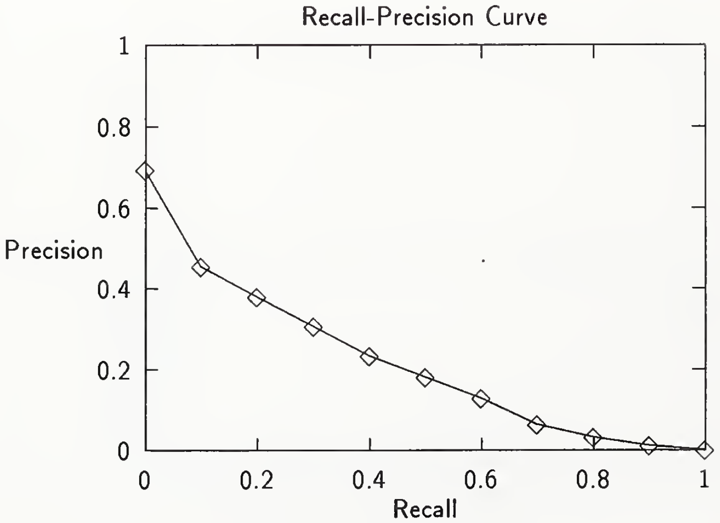
Recall Level Precision Averages	
Recall	Precision
0.00	0.6932
0.10	0.4543
0.20	0.3794
0.30	0.3061
0.40	0.2341
0.50	0.1817
0.60	0.1293
0.70	0.0640
0.80	0.0314
0.90	0.0116
1.00	0.0000

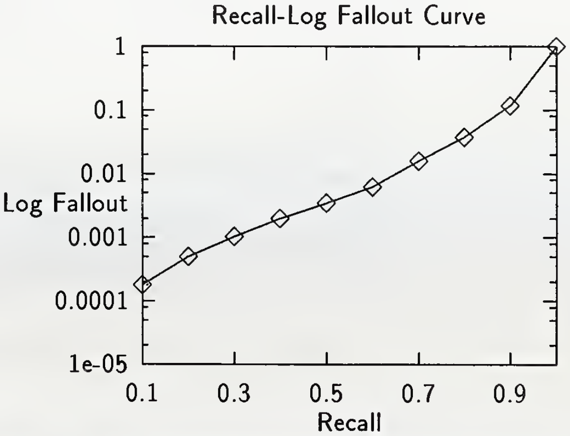
Average precision over all relevant docs	
non-interpolated	0.2047

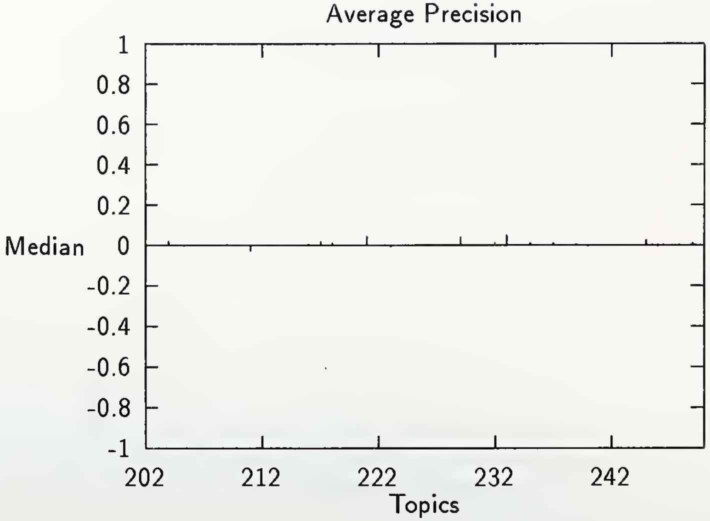
Document Level Averages	
	Precision
At 5 docs	0.5061
At 10 docs	0.4429
At 15 docs	0.4000
At 20 docs	0.3888
At 30 docs	0.3483
At 100 docs	0.2361
At 200 docs	0.1785
At 500 docs	0.1043
At 1000 docs	0.0634

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2603

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00018
0.20	0.00050
0.30	0.00104
0.40	0.00200
0.50	0.00345
0.60	0.00619
0.70	0.01568
0.80	0.03779
0.90	0.11742
1.00	1.00000







Summary Statistics	
Run Number	padreW–category A, automatic
Number of Topics	49

Total number of documents over all topics	
Retrieved:	31304
Relevant:	6501
Rel_ret:	3602

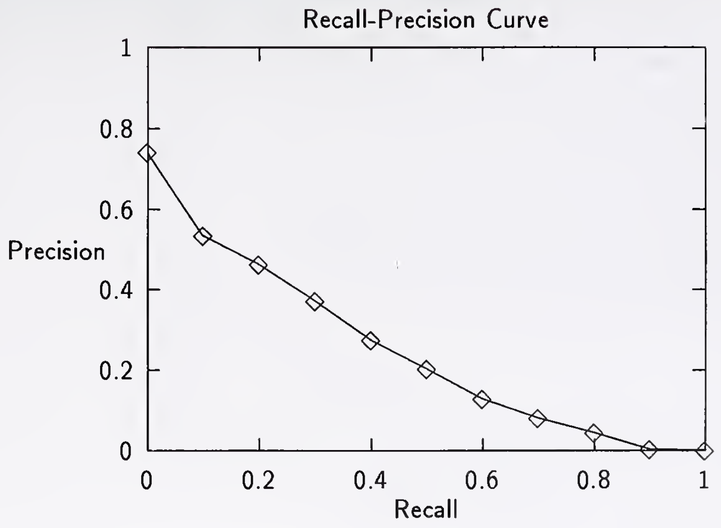
Recall Level Precision Averages	
Recall	Precision
0.00	0.7416
0.10	0.5342
0.20	0.4630
0.30	0.3719
0.40	0.2753
0.50	0.2042
0.60	0.1287
0.70	0.0818
0.80	0.0445
0.90	0.0026
1.00	0.0000

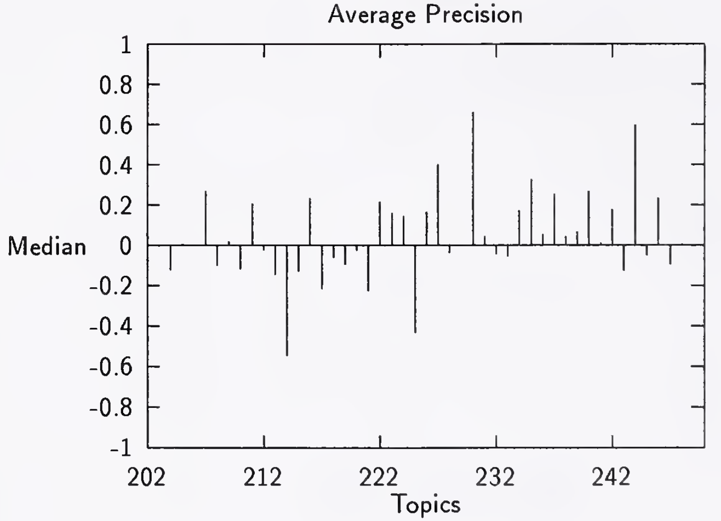
Average precision over all relevant docs	
non-interpolated	0.2383

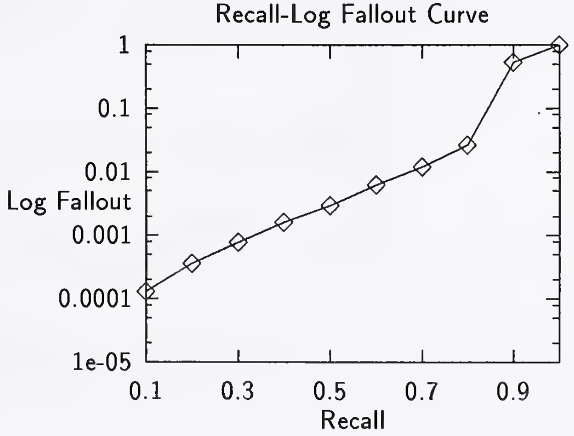
Document Level Averages	
	Precision
At 5 docs	0.5306
At 10 docs	0.5143
At 15 docs	0.4884
At 20 docs	0.4694
At 30 docs	0.4306
At 100 docs	0.2988
At 200 docs	0.2152
At 500 docs	0.1242
At 1000 docs	0.0735

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2934

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00013
0.20	0.00036
0.30	0.00078
0.40	0.00161
0.50	0.00298
0.60	0.00622
0.70	0.01203
0.80	0.02630
0.90	0.52866
1.00	1.00000





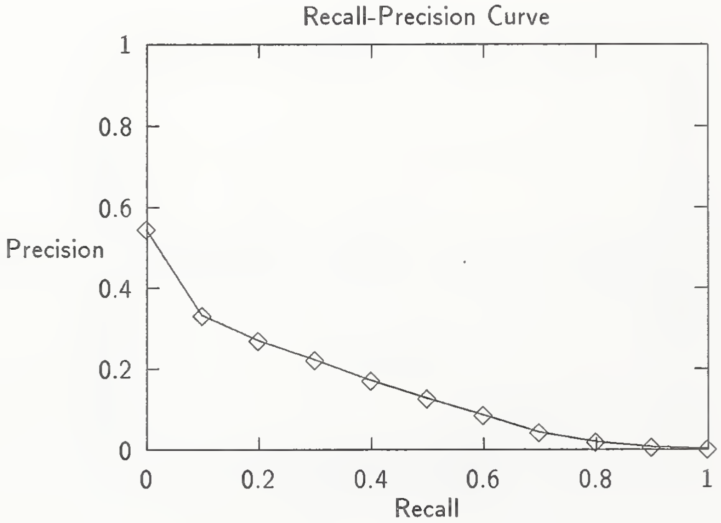


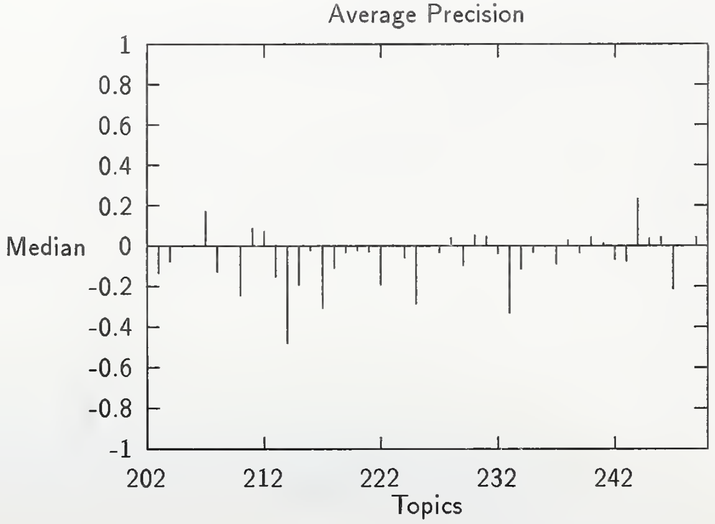
Summary Statistics	
Run Number	siems2–category A, automatic
Number of Topics	49
Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	3290

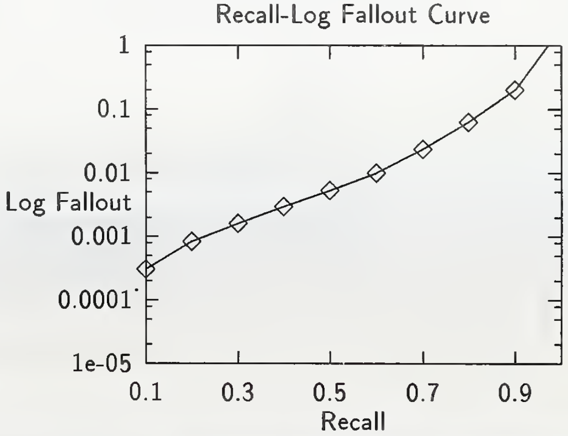
Recall Level Precision Averages	
Recall	Precision
0.00	0.5449
0.10	0.3326
0.20	0.2706
0.30	0.2223
0.40	0.1715
0.50	0.1272
0.60	0.0856
0.70	0.0433
0.80	0.0193
0.90	0.0069
1.00	0.0008
Average precision over all relevant docs	
non-interpolated	0.1433

Document Level Averages	
	Precision
At 5 docs	0.3061
At 10 docs	0.3102
At 15 docs	0.3020
At 20 docs	0.2949
At 30 docs	0.2741
At 100 docs	0.2167
At 200 docs	0.1692
At 500 docs	0.1055
At 1000 docs	0.0671
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2139

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00031
0.20	0.00083
0.30	0.00161
0.40	0.00296
0.50	0.00525
0.60	0.00981
0.70	0.02368
0.80	0.06225
0.90	0.19835
1.00	1.91251







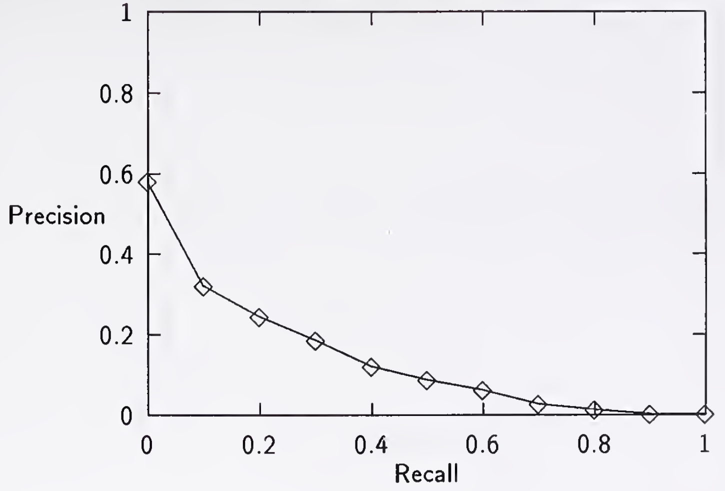
Summary Statistics	
Run Number	siems3–category A, automatic
Number of Topics	49
Total number of documents over all topics	
Retrieved:	49000
Relevant:	6501
Rel_ret:	2626

Recall Level Precision Averages	
Recall	Precision
0.00	0.5807
0.10	0.3210
0.20	0.2444
0.30	0.1863
0.40	0.1216
0.50	0.0871
0.60	0.0619
0.70	0.0273
0.80	0.0126
0.90	0.0013
1.00	0.0009
Average precision over all relevant docs	
non-interpolated	0.1253

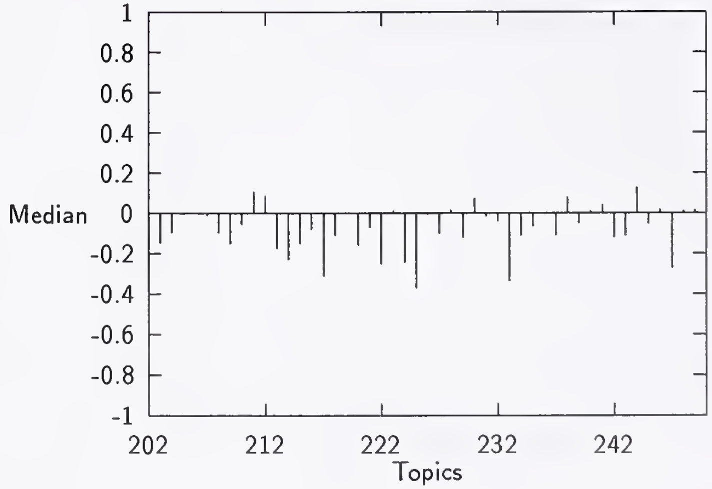
Document Level Averages	
	Precision
At 5 docs	0.3959
At 10 docs	0.3286
At 15 docs	0.2993
At 20 docs	0.2847
At 30 docs	0.2653
At 100 docs	0.2053
At 200 docs	0.1495
At 500 docs	0.0871
At 1000 docs	0.0536
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1934

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00032
0.20	0.00095
0.30	0.00201
0.40	0.00442
0.50	0.00802
0.60	0.01392
0.70	0.03819
0.80	0.09600
0.90	1.05871
1.00	1.69984

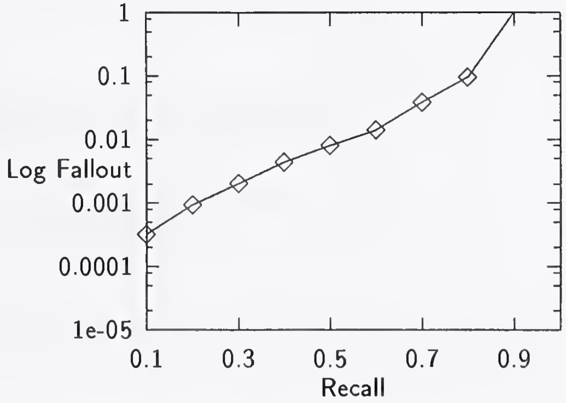
Recall-Precision Curve



Average Precision



Recall-Log Fallout Curve



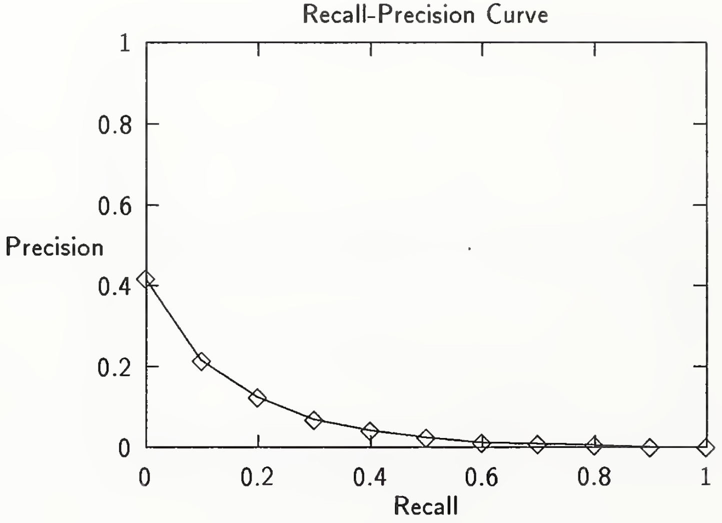
Summary Statistics	
Run Number	ACQSPA–category A, manual
Number of Topics	25

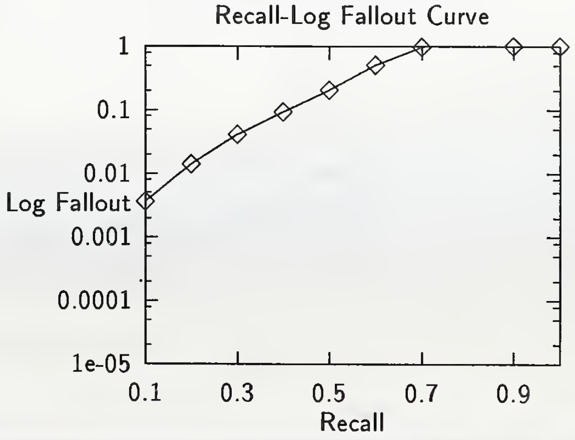
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2202
Rel_ret:	804

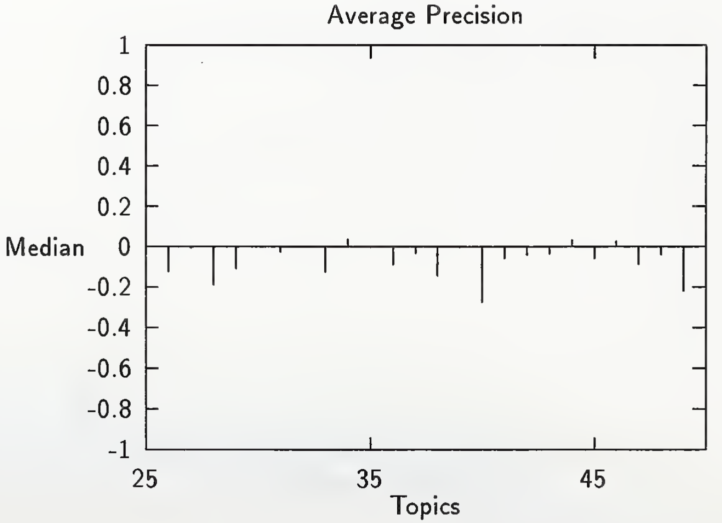
Recall Level Precision Averages	
Recall	Precision
0.00	0.4177
0.10	0.2145
0.20	0.1239
0.30	0.0676
0.40	0.0414
0.50	0.0238
0.60	0.0117
0.70	0.0072
0.80	0.0047
0.90	0.0000
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.0622

Document Level Averages	
	Precision
At 5 docs	0.1920
At 10 docs	0.2320
At 15 docs	0.2107
At 20 docs	0.1840
At 30 docs	0.1627
At 100 docs	0.1160
At 200 docs	0.0850
At 500 docs	0.0494
At 1000 docs	0.0322
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1175

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00367
0.20	0.01417
0.30	0.04145
0.40	0.09278
0.50	0.20545
0.60	0.50772
0.70	0.96693
0.80	1.69713
0.90	1.00000
1.00	1.00000







Summary Statistics	
Run Number	BrklySP3–category A, automatic
Number of Topics	25

Total number of documents over all topics	
Retrieved:	25000
Relevant:	2202
Rel_ret:	1309

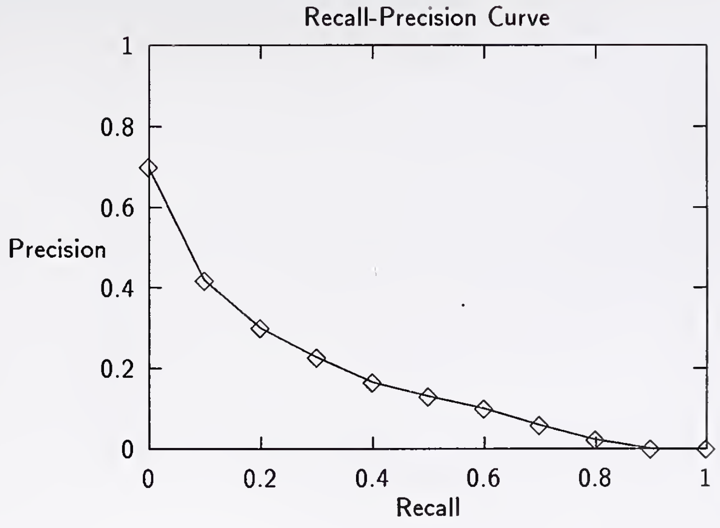
Recall Level Precision Averages	
Recall	Precision
0.00	0.6994
0.10	0.4180
0.20	0.3011
0.30	0.2278
0.40	0.1661
0.50	0.1303
0.60	0.1006
0.70	0.0587
0.80	0.0224
0.90	0.0000
1.00	0.0000

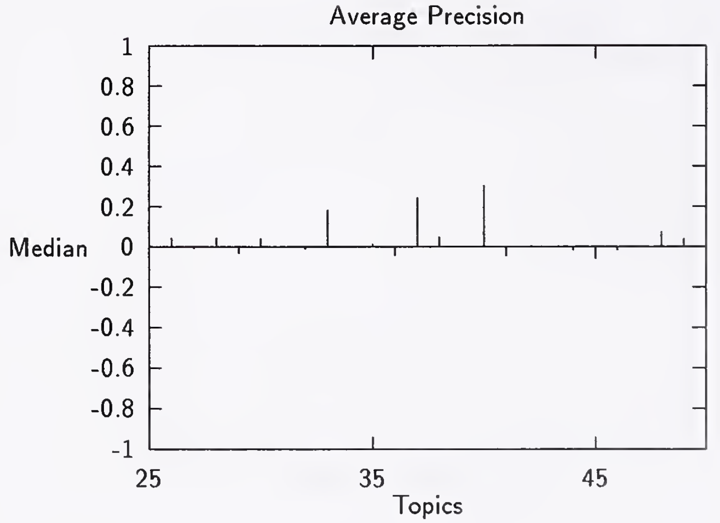
Average precision over all relevant docs	
non-interpolated	0.1683

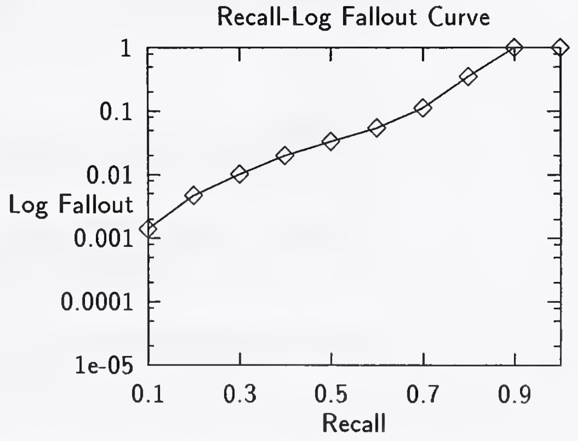
Document Level Averages	
	Precision
At 5 docs	0.4800
At 10 docs	0.4080
At 15 docs	0.3813
At 20 docs	0.3480
At 30 docs	0.3120
At 100 docs	0.1968
At 200 docs	0.1448
At 500 docs	0.0818
At 1000 docs	0.0524

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2188

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00139
0.20	0.00465
0.30	0.01019
0.40	0.02012
0.50	0.03343
0.60	0.05374
0.70	0.11245
0.80	0.34976
0.90	1.00000
1.00	1.00000







Summary Statistics	
Run Number	BrklySP4–category A, manual
Number of Topics	25

Total number of documents over all topics	
Retrieved:	25000
Relevant:	2202
Rel_ret:	1310

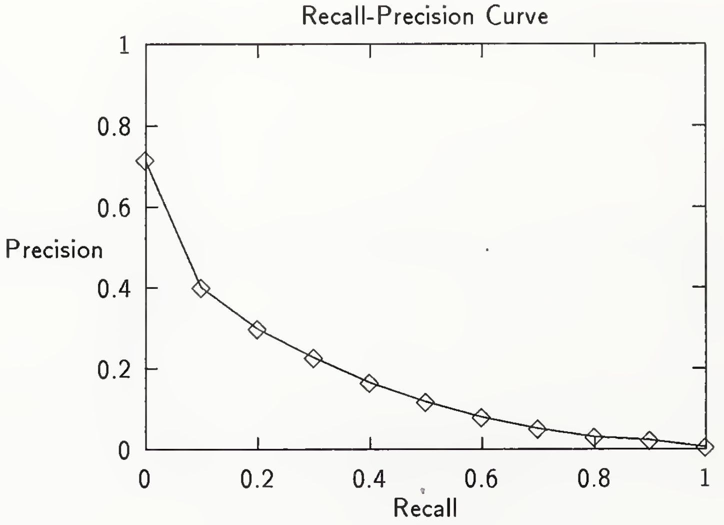
Recall Level Precision Averages	
Recall	Precision
0.00	0.7148
0.10	0.4008
0.20	0.2992
0.30	0.2268
0.40	0.1652
0.50	0.1184
0.60	0.0792
0.70	0.0516
0.80	0.0299
0.90	0.0230
1.00	0.0044

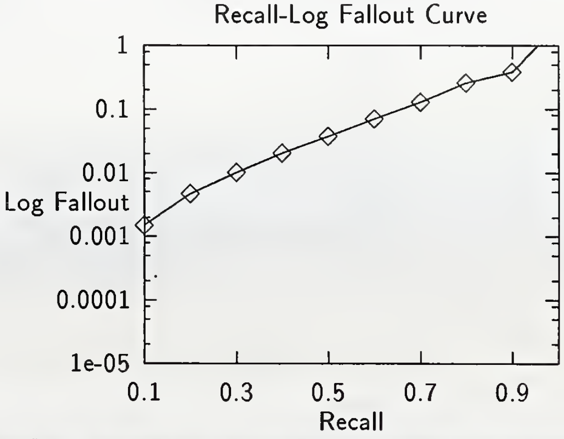
Average precision over all relevant docs	
non-interpolated	0.1637

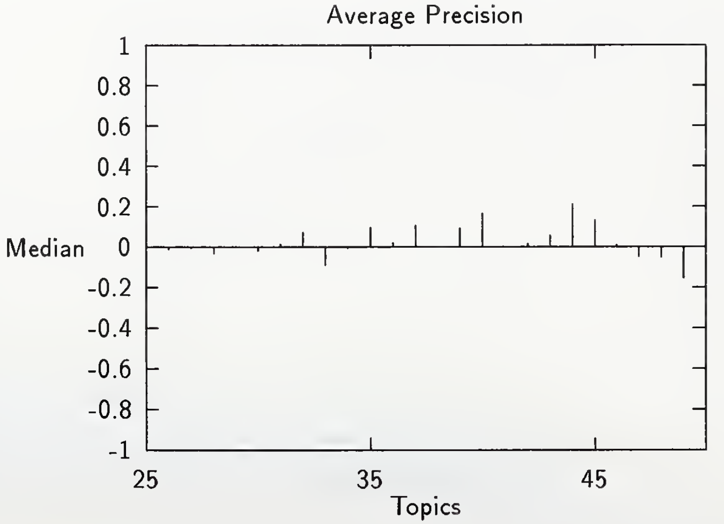
Document Level Averages	
	Precision
At 5 docs	0.4720
At 10 docs	0.4200
At 15 docs	0.3867
At 20 docs	0.3460
At 30 docs	0.3160
At 100 docs	0.2240
At 200 docs	0.1556
At 500 docs	0.0874
At 1000 docs	0.0524

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2344

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00150
0.20	0.00469
0.30	0.01025
0.40	0.02025
0.50	0.03730
0.60	0.06988
0.70	0.12889
0.80	0.26002
0.90	0.38298
1.00	2.26674







Summary Statistics	
Run Number	citri-sp1–category A, automatic
Number of Topics	25

Total number of documents over all topics	
Retrieved:	25000
Relevant:	2202
Rel_ret:	1285

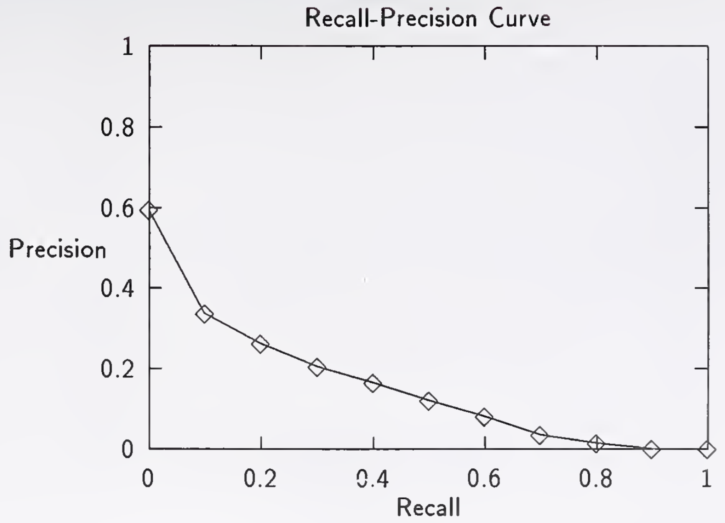
Recall Level Precision Averages	
Recall	Precision
0.00	0.5952
0.10	0.3369
0.20	0.2630
0.30	0.2057
0.40	0.1651
0.50	0.1209
0.60	0.0808
0.70	0.0353
0.80	0.0137
0.90	0.0000
1.00	0.0000

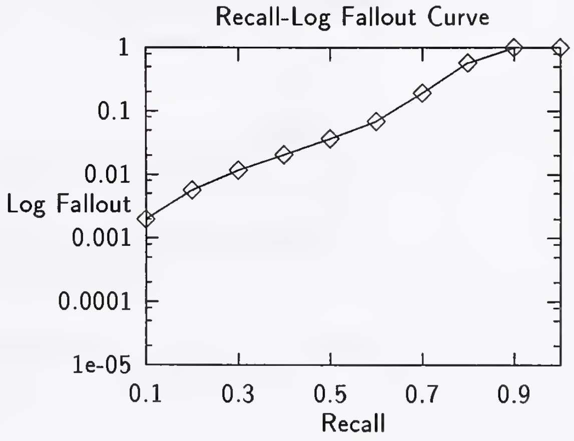
Average precision over all relevant docs	
non-interpolated	0.1427

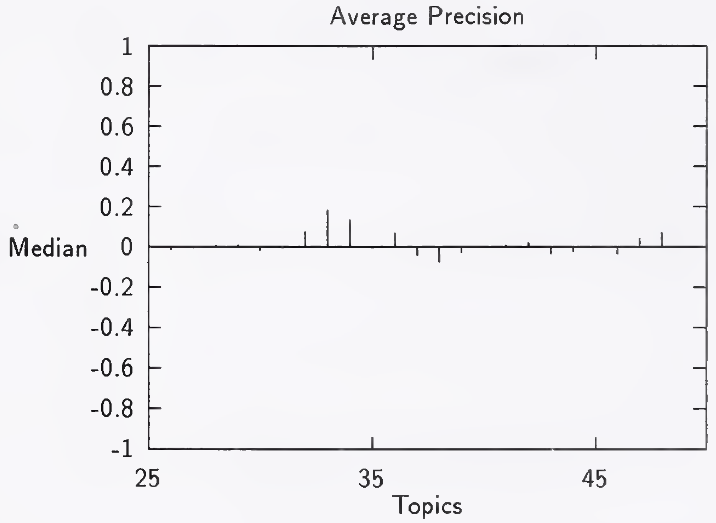
Document Level Averages	
	Precision
At 5 docs	0.3840
At 10 docs	0.3680
At 15 docs	0.3253
At 20 docs	0.3200
At 30 docs	0.2840
At 100 docs	0.2000
At 200 docs	0.1386
At 500 docs	0.0808
At 1000 docs	0.0514

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2025

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00197
0.20	0.00561
0.30	0.01160
0.40	0.02026
0.50	0.03642
0.60	0.06838
0.70	0.19164
0.80	0.57696
0.90	1.00000
1.00	1.00000







Summary Statistics	
Run Number	citri-sp2–category A, automatic
Number of Topics	25

Total number of documents over all topics	
Retrieved:	25000
Relevant:	2202
Rel_ret:	1379

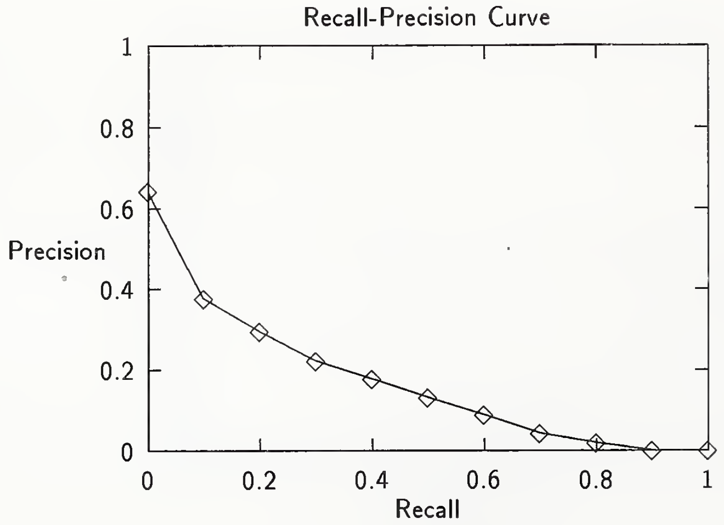
Recall Level Precision Averages	
Recall	Precision
0.00	0.6417
0.10	0.3771
0.20	0.2954
0.30	0.2221
0.40	0.1784
0.50	0.1323
0.60	0.0894
0.70	0.0432
0.80	0.0192
0.90	0.0006
1.00	0.0000

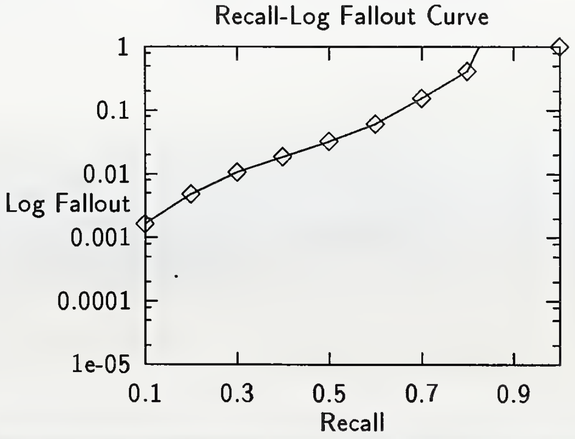
Average precision over all relevant docs	
non-interpolated	0.1582

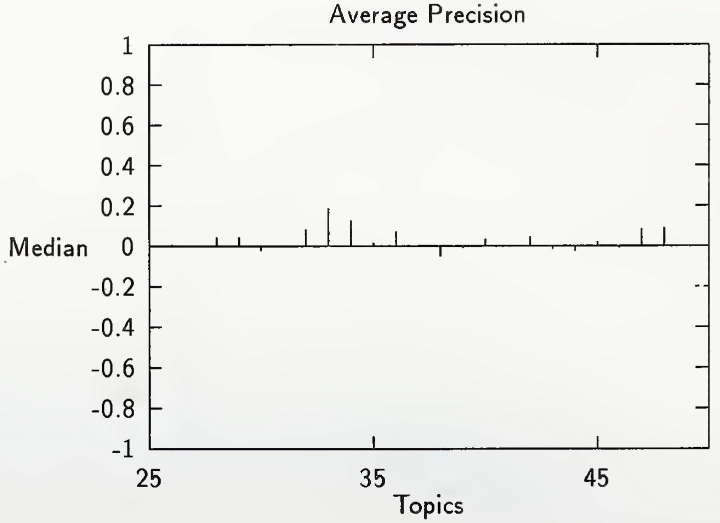
Document Level Averages	
	Precision
At 5 docs	0.3920
At 10 docs	0.3960
At 15 docs	0.3680
At 20 docs	0.3500
At 30 docs	0.3200
At 100 docs	0.2208
At 200 docs	0.1490
At 500 docs	0.0875
At 1000 docs	0.0552

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2218

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00165
0.20	0.00478
0.30	0.01053
0.40	0.01845
0.50	0.03285
0.60	0.06122
0.70	0.15531
0.80	0.40939
0.90	15.01759
1.00	1.00000







Summary Statistics	
Run Number	CrnlSE–category A, automatic
Number of Topics	25

Total number of documents over all topics	
Retrieved:	25000
Relevant:	2202
Rel_ret:	1664

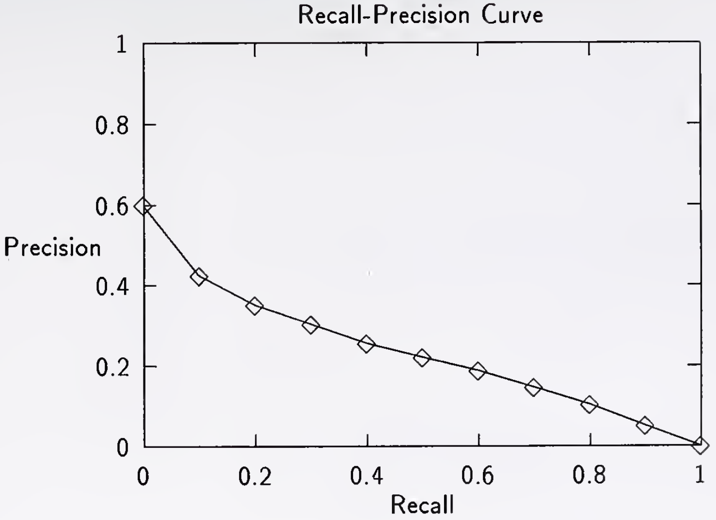
Recall Level Precision Averages	
Recall	Precision
0.00	0.5999
0.10	0.4248
0.20	0.3508
0.30	0.3040
0.40	0.2558
0.50	0.2213
0.60	0.1883
0.70	0.1468
0.80	0.1038
0.90	0.0505
1.00	0.0000

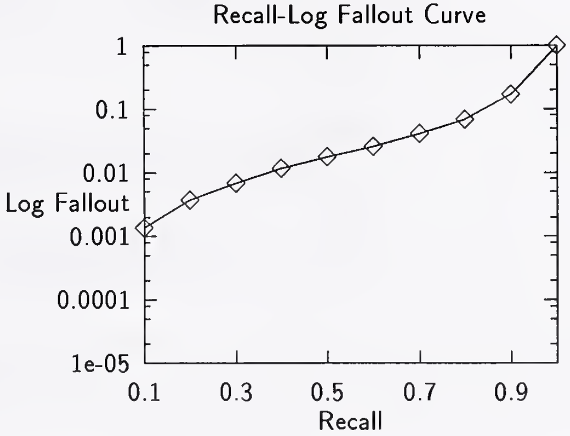
Average precision over all relevant docs	
non-interpolated	0.2234

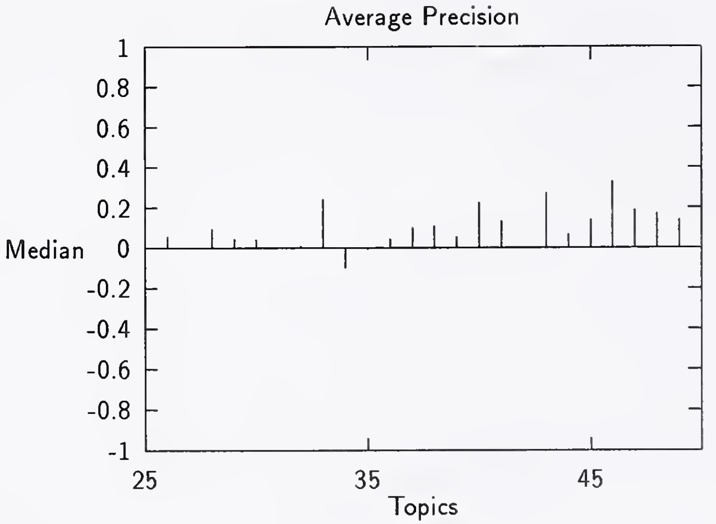
Document Level Averages	
	Precision
At 5 docs	0.3680
At 10 docs	0.3960
At 15 docs	0.3680
At 20 docs	0.3580
At 30 docs	0.3360
At 100 docs	0.2684
At 200 docs	0.2000
At 500 docs	0.1122
At 1000 docs	0.0666

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2807

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00136
0.20	0.00371
0.30	0.00688
0.40	0.01166
0.50	0.01762
0.60	0.02591
0.70	0.04076
0.80	0.06919
0.90	0.16952
1.00	1.00000







Summary Statistics	
Run Number	DCUSP0–category A, manual
Number of Topics	25

Total number of documents over all topics	
Retrieved:	24986
Relevant:	2202
Rel_ret:	809

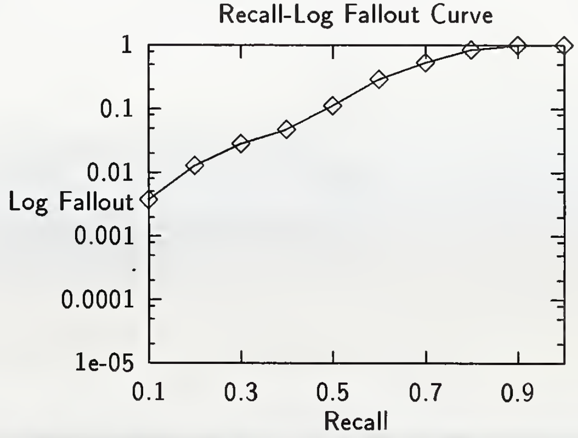
Recall Level Precision Averages	
Recall	Precision
0.00	0.4066
0.10	0.2120
0.20	0.1356
0.30	0.0967
0.40	0.0775
0.50	0.0427
0.60	0.0202
0.70	0.0129
0.80	0.0093
0.90	0.0000
1.00	0.0000

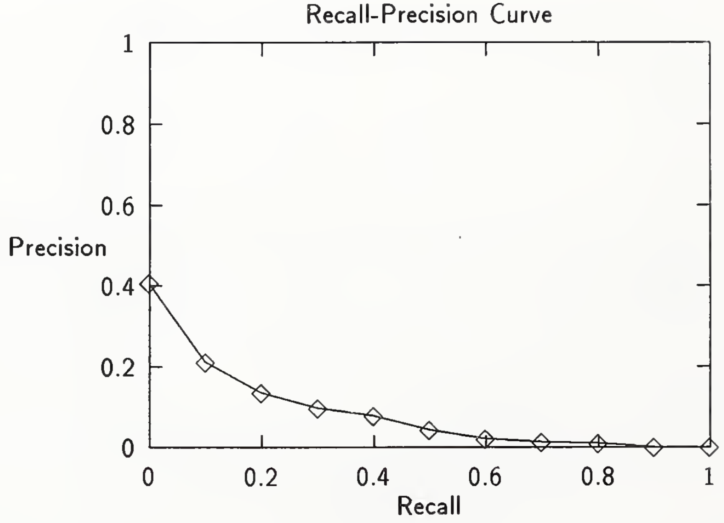
Average precision over all relevant docs	
non-interpolated	0.0735

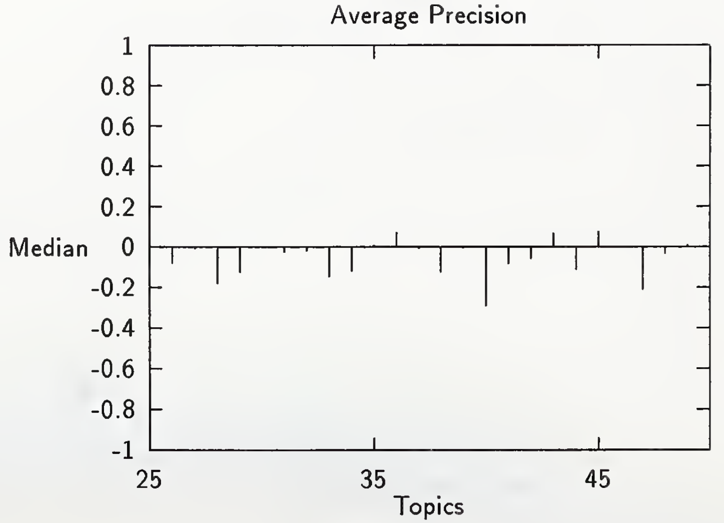
Document Level Averages	
	Precision
At 5 docs	0.1760
At 10 docs	0.1840
At 15 docs	0.1787
At 20 docs	0.1700
At 30 docs	0.1560
At 100 docs	0.1136
At 200 docs	0.0804
At 500 docs	0.0498
At 1000 docs	0.0324

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1214

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00372
0.20	0.01277
0.30	0.02807
0.40	0.04770
0.50	0.11229
0.60	0.29155
0.70	0.53659
0.80	0.85373
0.90	1.00000
1.00	1.00000







Summary Statistics	
Run Number	gmuauto–category A, automatic
Number of Topics	25

Total number of documents over all topics	
Retrieved:	25000
Relevant:	2202
Rel_ret:	1504

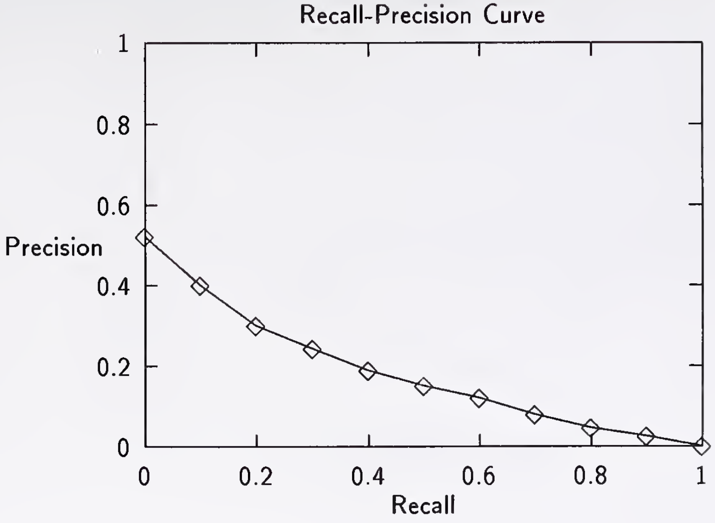
Recall Level Precision Averages	
Recall	Precision
0.00	0.5215
0.10	0.4010
0.20	0.3003
0.30	0.2431
0.40	0.1896
0.50	0.1512
0.60	0.1203
0.70	0.0801
0.80	0.0463
0.90	0.0258
1.00	0.0000

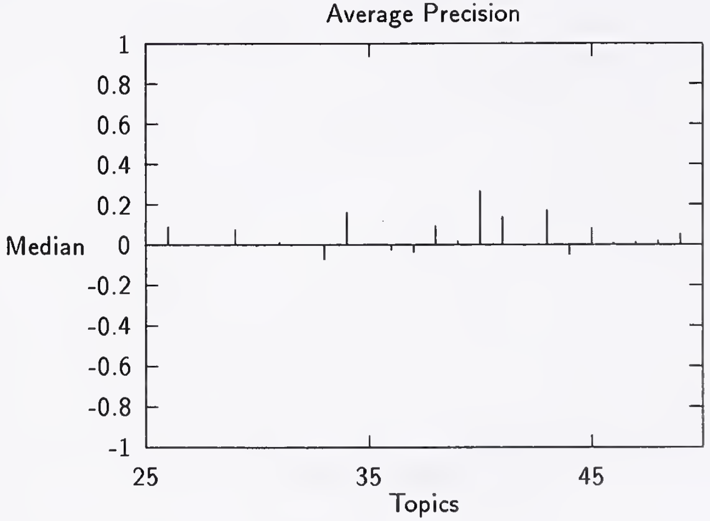
Average precision over all relevant docs	
non-interpolated	0.1722

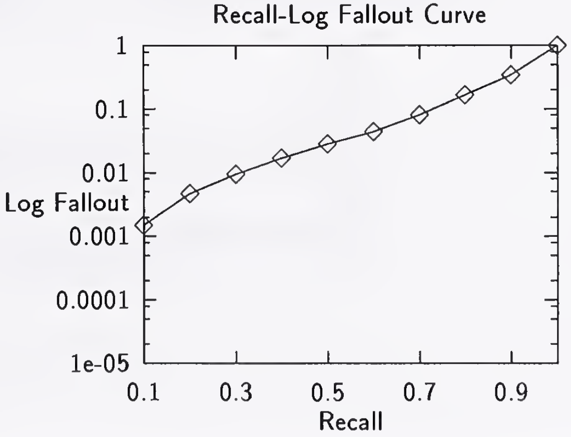
Document Level Averages	
	Precision
At 5 docs	0.3360
At 10 docs	0.3520
At 15 docs	0.3467
At 20 docs	0.3280
At 30 docs	0.3107
At 100 docs	0.2312
At 200 docs	0.1676
At 500 docs	0.0973
At 1000 docs	0.0602

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2385

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00150
0.20	0.00467
0.30	0.00936
0.40	0.01713
0.50	0.02812
0.60	0.04395
0.70	0.08053
0.80	0.16508
0.90	0.34044
1.00	1.00000





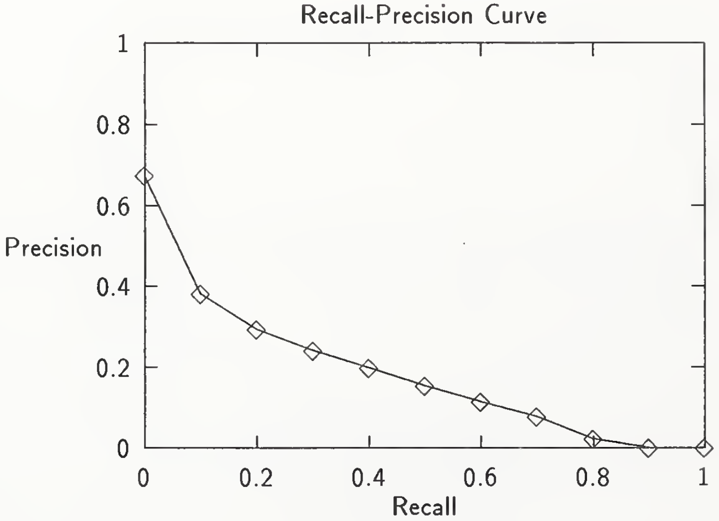


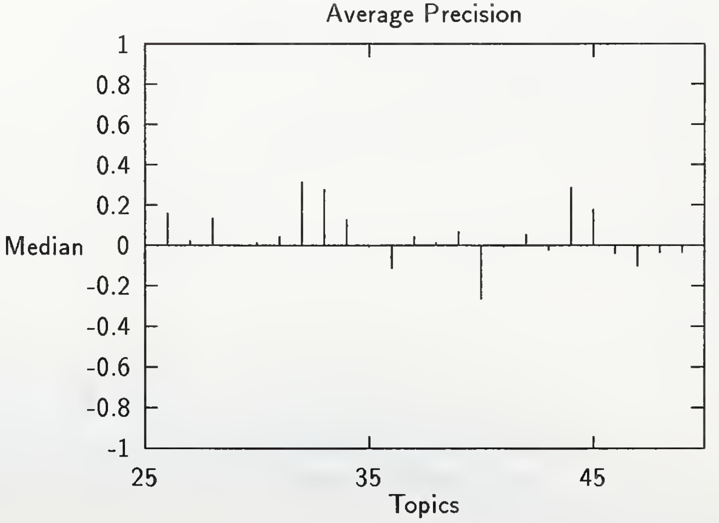
Summary Statistics	
Run Number	gmuman–category A, manual
Number of Topics	25
Total number of documents over all topics	
Retrieved:	16230
Relevant:	2202
Rel_ret:	1185

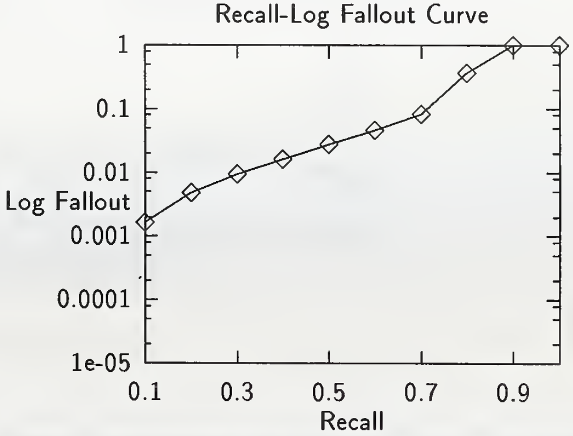
Recall Level Precision Averages	
Recall	Precision
0.00	0.6735
0.10	0.3810
0.20	0.2938
0.30	0.2415
0.40	0.1995
0.50	0.1537
0.60	0.1144
0.70	0.0782
0.80	0.0216
0.90	0.0000
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.1708

Document Level Averages	
	Precision
At 5 docs	0.4560
At 10 docs	0.3760
At 15 docs	0.3627
At 20 docs	0.3440
At 30 docs	0.3160
At 100 docs	0.2444
At 200 docs	0.1658
At 500 docs	0.0862
At 1000 docs	0.0474
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2387

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00163
0.20	0.00482
0.30	0.00944
0.40	0.01608
0.50	0.02758
0.60	0.04653
0.70	0.08266
0.80	0.36301
0.90	1.00000
1.00	1.00000







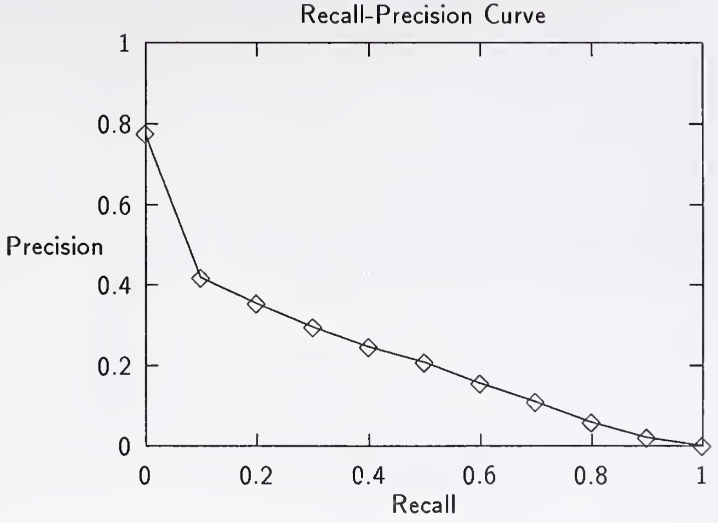
Summary Statistics	
Run Number	nmsu0–category A, automatic
Number of Topics	25

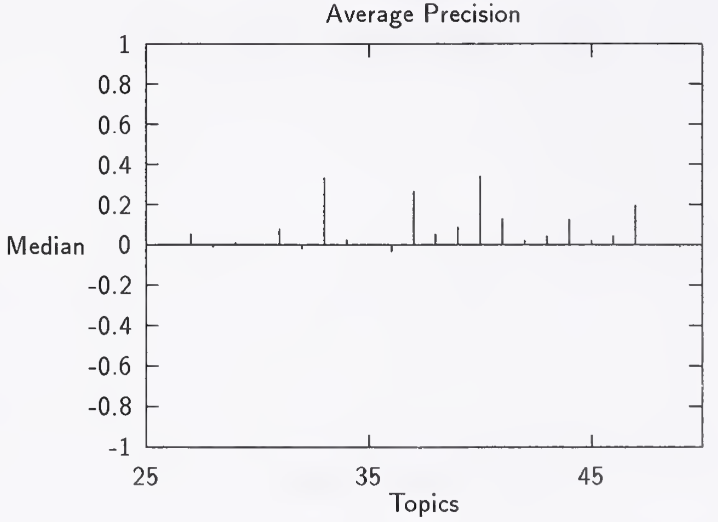
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2202
Rel_ret:	1521

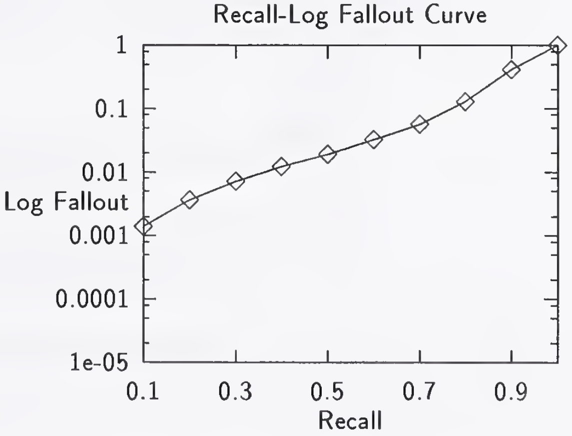
Recall Level Precision Averages	
Recall	Precision
0.00	0.7763
0.10	0.4189
0.20	0.3546
0.30	0.2954
0.40	0.2459
0.50	0.2081
0.60	0.1551
0.70	0.1104
0.80	0.0589
0.90	0.0213
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.2153

Document Level Averages	
	Precision
At 5 docs	0.4880
At 10 docs	0.4400
At 15 docs	0.4000
At 20 docs	0.3900
At 30 docs	0.3573
At 100 docs	0.2452
At 200 docs	0.1758
At 500 docs	0.0999
At 1000 docs	0.0608
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2646

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00139
0.20	0.00365
0.30	0.00717
0.40	0.01229
0.50	0.01906
0.60	0.03274
0.70	0.05651
0.80	0.12805
0.90	0.41427
1.00	1.00000







Summary Statistics	
Run Number	nmsu1–category A, manual
Number of Topics	25
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2202
Rel_ret:	663

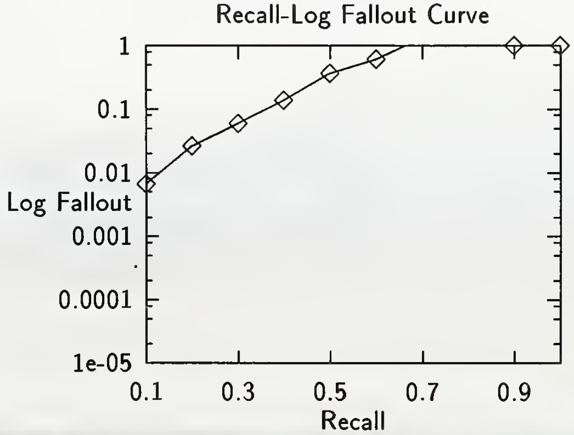
Recall Level Precision Averages	
Recall	Precision
0.00	0.3322
0.10	0.1315
0.20	0.0708
0.30	0.0483
0.40	0.0285
0.50	0.0135
0.60	0.0098
0.70	0.0052
0.80	0.0043
0.90	0.0000
1.00	0.0000

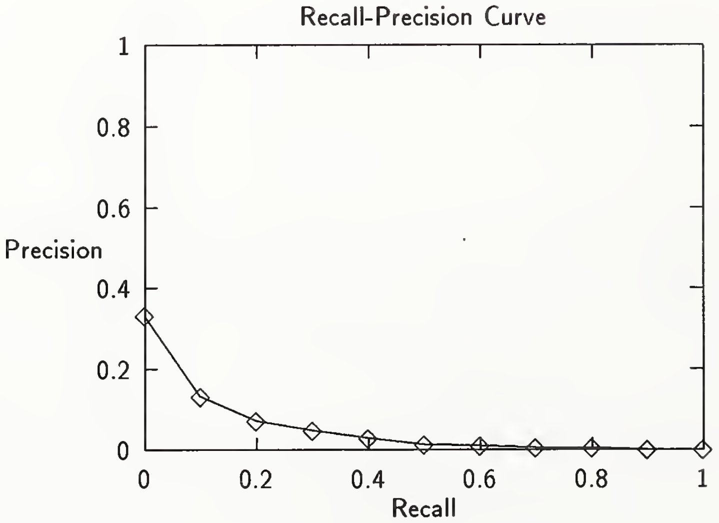
Average precision over all relevant docs	
non-interpolated	0.0414

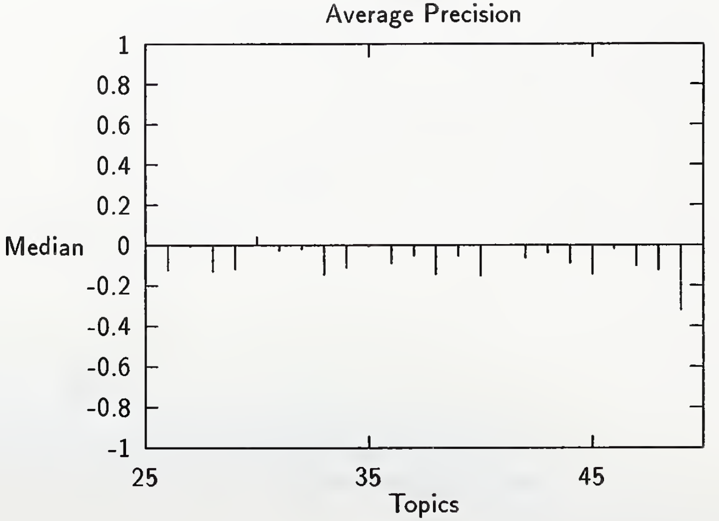
Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00662
0.20	0.02630
0.30	0.05922
0.40	0.13659
0.50	0.36602
0.60	0.60732
0.70	1.34153
0.80	1.85575
0.90	1.00000
1.00	1.00000

Document Level Averages	
	Precision
At 5 docs	0.1760
At 10 docs	0.1560
At 15 docs	0.1413
At 20 docs	0.1240
At 30 docs	0.1093
At 100 docs	0.0800
At 200 docs	0.0586
At 500 docs	0.0394
At 1000 docs	0.0265

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0837





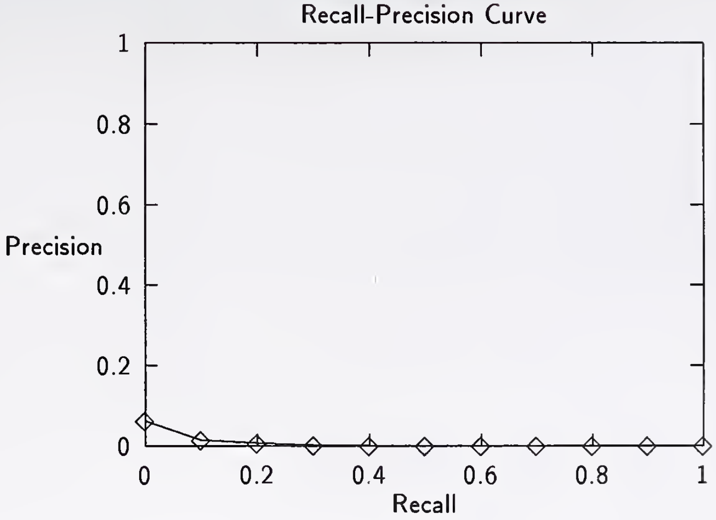


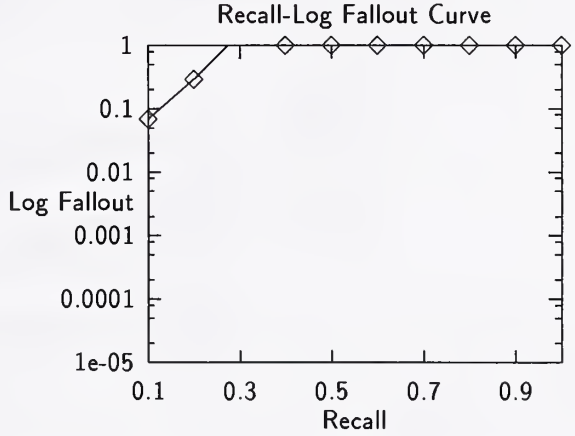
Summary Statistics	
Run Number	nmsu2–category A, manual
Number of Topics	25
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2202
Rel_ret:	280

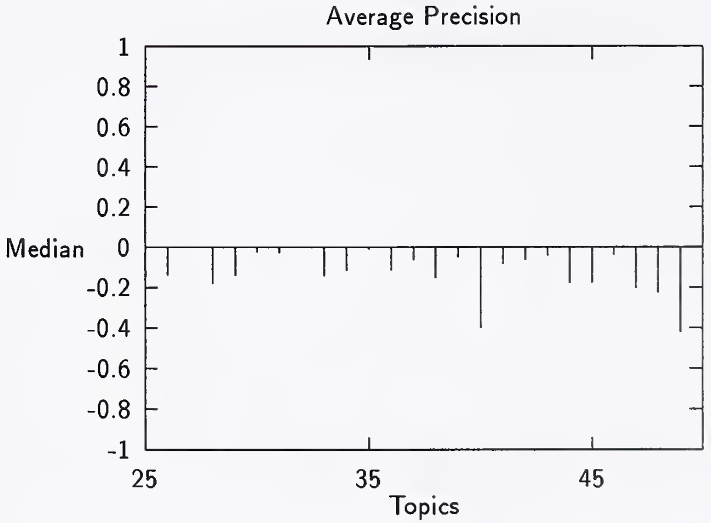
Recall Level Precision Averages	
Recall	Precision
0.00	0.0617
0.10	0.0144
0.20	0.0068
0.30	0.0020
0.40	0.0000
0.50	0.0000
0.60	0.0000
0.70	0.0000
0.80	0.0000
0.90	0.0000
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.0042

Document Level Averages	
	Precision
At 5 docs	0.0080
At 10 docs	0.0200
At 15 docs	0.0187
At 20 docs	0.0240
At 30 docs	0.0240
At 100 docs	0.0212
At 200 docs	0.0172
At 500 docs	0.0134
At 1000 docs	0.0112
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0240

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.06857
0.20	0.29264
0.30	1.49966
0.40	1.00000
0.50	1.00000
0.60	1.00000
0.70	1.00000
0.80	1.00000
0.90	1.00000
1.00	1.00000







Summary Statistics	
Run Number	nmsu3–category A, manual
Number of Topics	25

Total number of documents over all topics	
Retrieved:	25000
Relevant:	2202
Rel_ret:	410

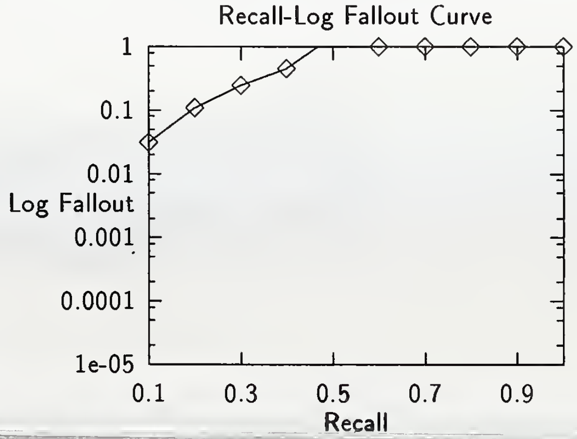
Recall Level Precision Averages	
Recall	Precision
0.00	0.1149
0.10	0.0314
0.20	0.0181
0.30	0.0120
0.40	0.0088
0.50	0.0034
0.60	0.0000
0.70	0.0000
0.80	0.0000
0.90	0.0000
1.00	0.0000

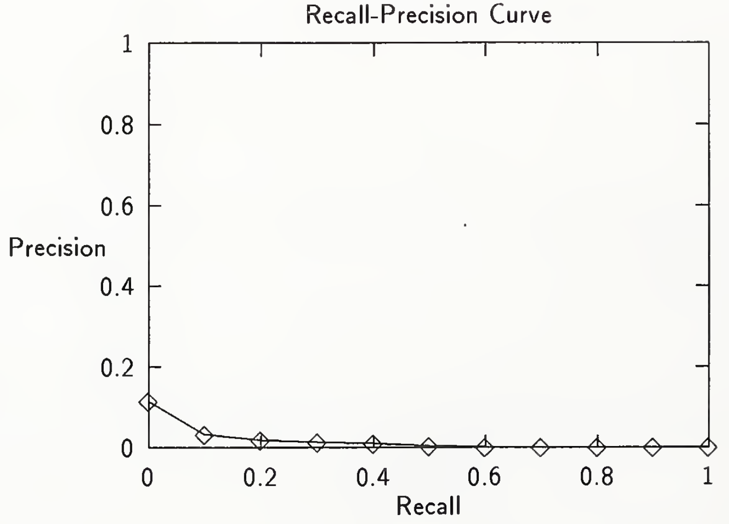
Average precision over all relevant docs	
non-interpolated	0.0109

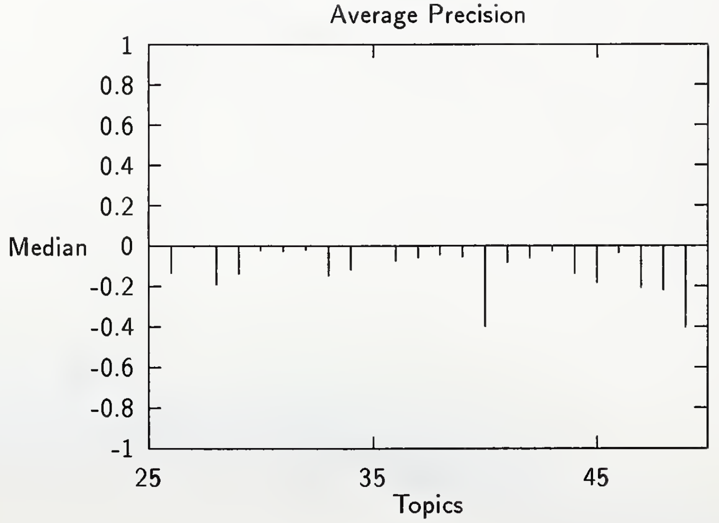
Document Level Averages	
	Precision
At 5 docs	0.0480
At 10 docs	0.0400
At 15 docs	0.0560
At 20 docs	0.0560
At 30 docs	0.0467
At 100 docs	0.0304
At 200 docs	0.0270
At 500 docs	0.0212
At 1000 docs	0.0164

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0340

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.03090
0.20	0.10869
0.30	0.24744
0.40	0.45134
0.50	1.46819
0.60	1.00000
0.70	1.00000
0.80	1.00000
0.90	1.00000
1.00	1.00000







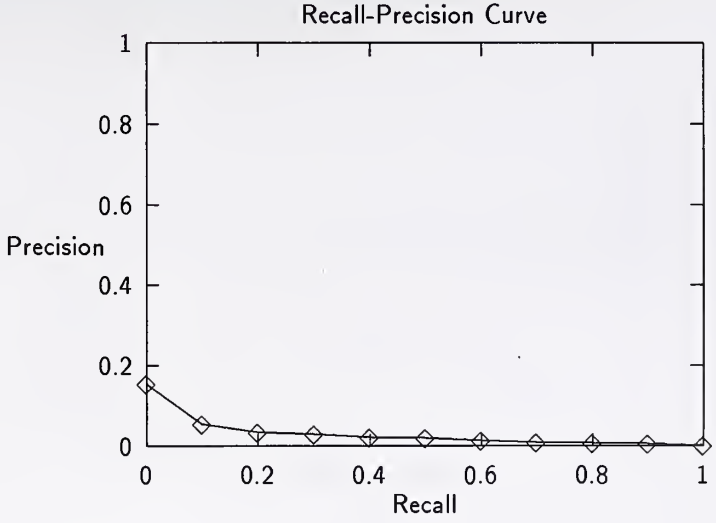
Summary Statistics	
Run Number	nmsu4–category A, manual
Number of Topics	25

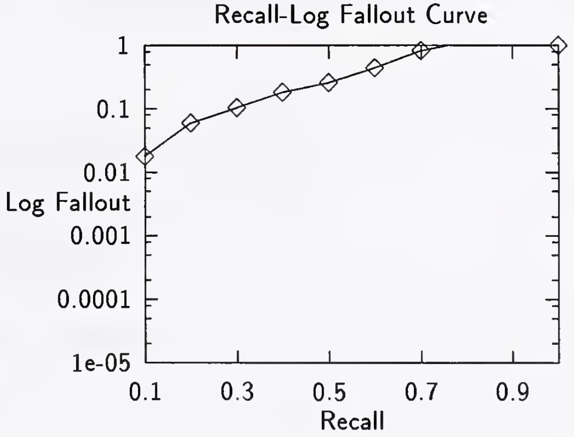
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2202
Rel_ret:	493

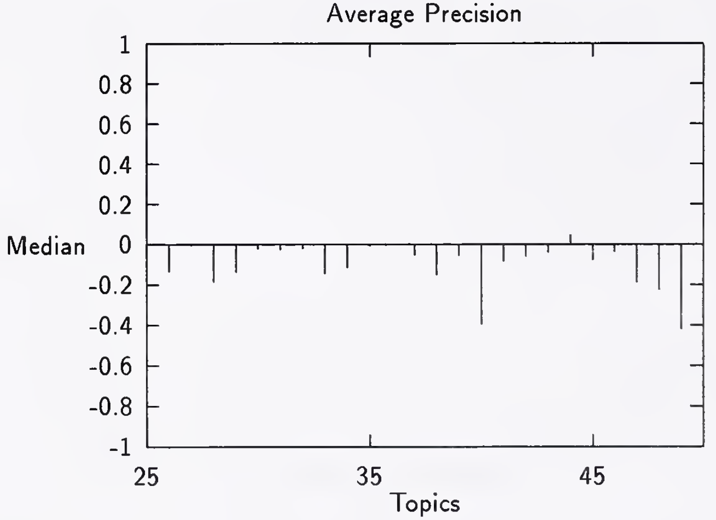
Recall Level Precision Averages	
Recall	Precision
0.00	0.1545
0.10	0.0538
0.20	0.0328
0.30	0.0280
0.40	0.0213
0.50	0.0190
0.60	0.0135
0.70	0.0084
0.80	0.0069
0.90	0.0050
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.0226

Document Level Averages	
	Precision
At 5 docs	0.0400
At 10 docs	0.0440
At 15 docs	0.0507
At 20 docs	0.0540
At 30 docs	0.0520
At 100 docs	0.0408
At 200 docs	0.0364
At 500 docs	0.0272
At 1000 docs	0.0197
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0420

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.01762
0.20	0.05908
0.30	0.10433
0.40	0.18412
0.50	0.25862
0.60	0.43922
0.70	0.82780
0.80	1.15346
0.90	1.79418
1.00	1.00000







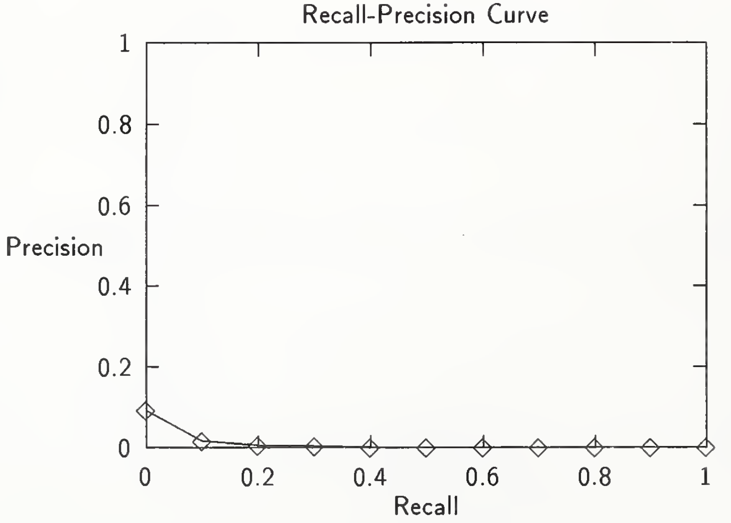
Summary Statistics	
Run Number	nmsu5–category A, manual
Number of Topics	25

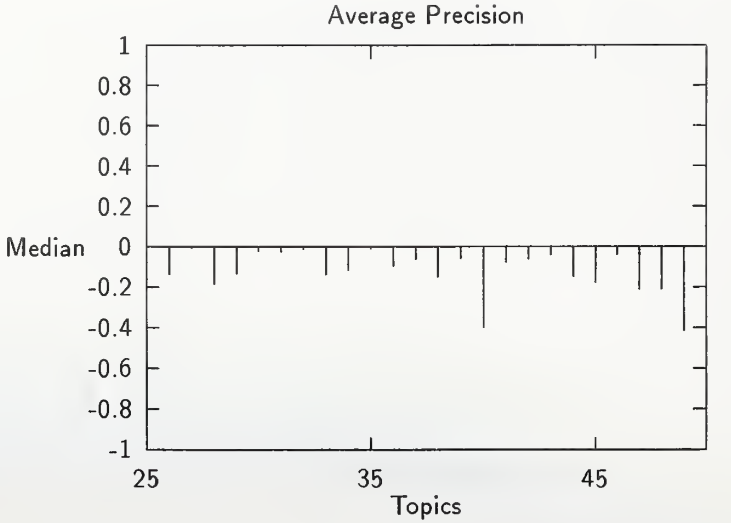
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2202
Rel_ret:	292

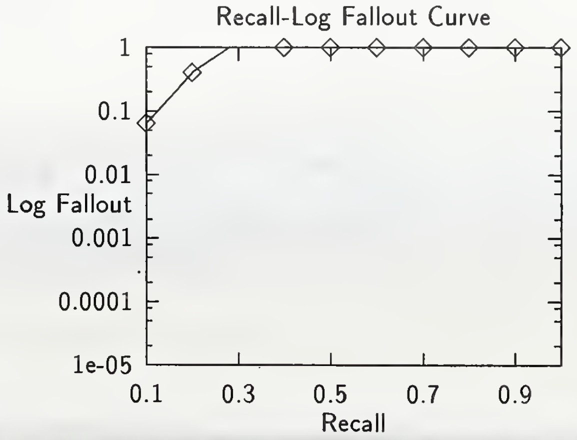
Recall Level Precision Averages	
Recall	Precision
0.00	0.0917
0.10	0.0155
0.20	0.0050
0.30	0.0024
0.40	0.0000
0.50	0.0000
0.60	0.0000
0.70	0.0000
0.80	0.0000
0.90	0.0000
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.0049

Document Level Averages	
	Precision
At 5 docs	0.0320
At 10 docs	0.0360
At 15 docs	0.0320
At 20 docs	0.0340
At 30 docs	0.0333
At 100 docs	0.0192
At 200 docs	0.0166
At 500 docs	0.0151
At 1000 docs	0.0117
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0201

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.06363
0.20	0.39871
0.30	1.24921
0.40	1.00000
0.50	1.00000
0.60	1.00000
0.70	1.00000
0.80	1.00000
0.90	1.00000
1.00	1.00000







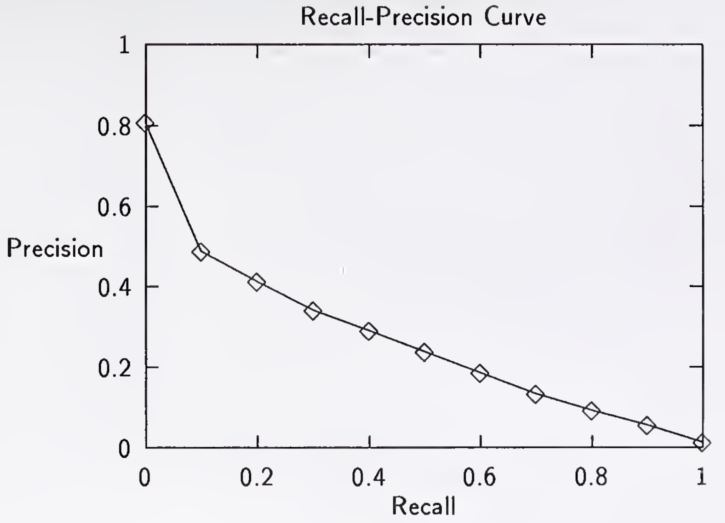
Summary Statistics	
Run Number	SIN010–category A, automatic
Number of Topics	25

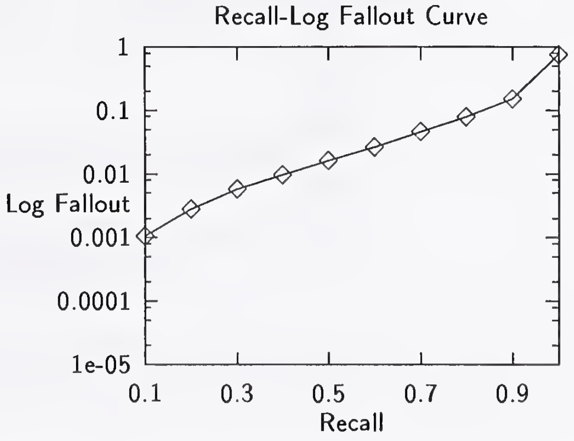
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2202
Rel_ret:	1604

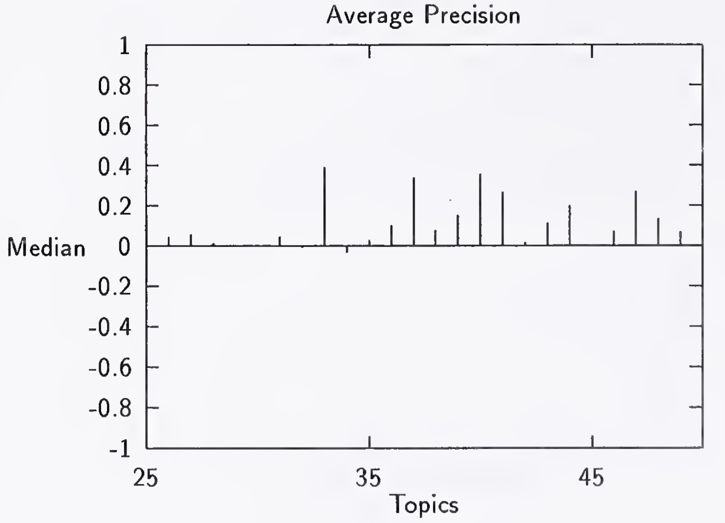
Recall Level Precision Averages	
Recall	Precision
0.00	0.8069
0.10	0.4887
0.20	0.4140
0.30	0.3422
0.40	0.2915
0.50	0.2380
0.60	0.1859
0.70	0.1330
0.80	0.0928
0.90	0.0558
1.00	0.0132
Average precision over all relevant docs	
non-interpolated	0.2523

Document Level Averages	
	Precision
At 5 docs	0.5040
At 10 docs	0.4800
At 15 docs	0.4480
At 20 docs	0.4280
At 30 docs	0.4000
At 100 docs	0.2804
At 200 docs	0.1966
At 500 docs	0.1071
At 1000 docs	0.0642
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2956

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00105
0.20	0.00284
0.30	0.00578
0.40	0.00974
0.50	0.01604
0.60	0.02632
0.70	0.04571
0.80	0.07835
0.90	0.15256
1.00	0.74890







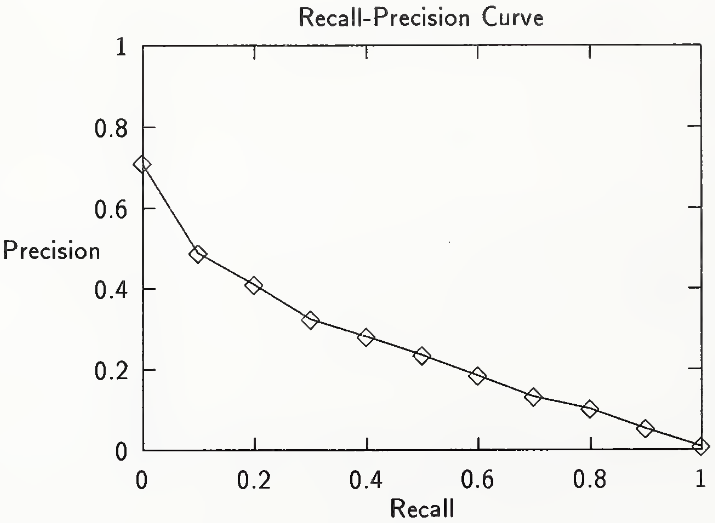
Summary Statistics	
Run Number	SIN011–category A, automatic
Number of Topics	25

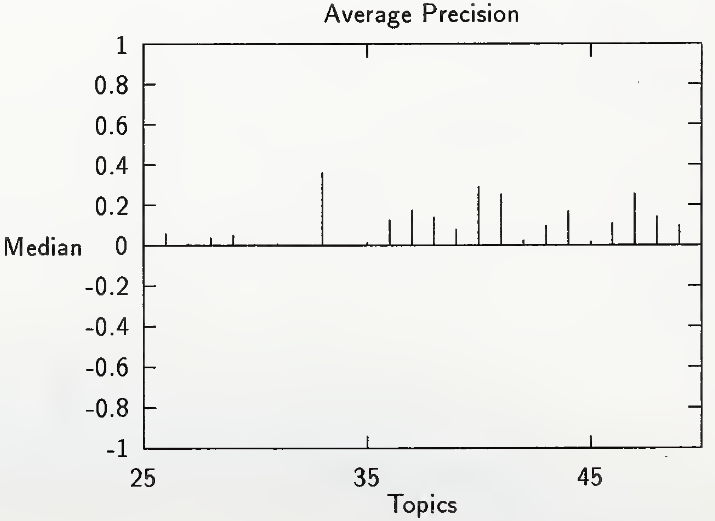
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2202
Rel_ret:	1644

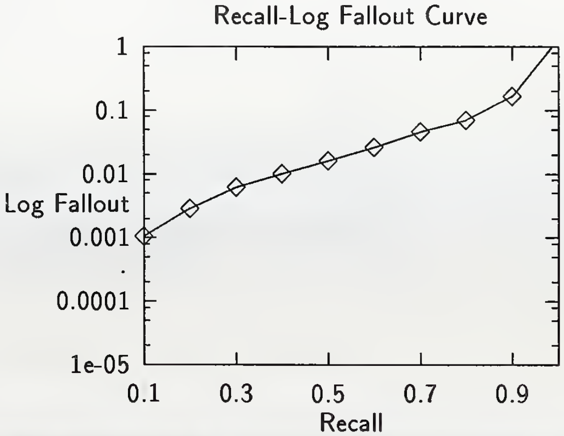
Recall Level Precision Averages	
Recall	Precision
0.00	0.7108
0.10	0.4880
0.20	0.4102
0.30	0.3243
0.40	0.2817
0.50	0.2354
0.60	0.1847
0.70	0.1312
0.80	0.1017
0.90	0.0519
1.00	0.0074
Average precision over all relevant docs	
non-interpolated	0.2458

Document Level Averages	
	Precision
At 5 docs	0.5040
At 10 docs	0.5040
At 15 docs	0.4693
At 20 docs	0.4300
At 30 docs	0.3880
At 100 docs	0.2760
At 200 docs	0.2008
At 500 docs	0.1111
At 1000 docs	0.0658
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2900

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00105
0.20	0.00288
0.30	0.00626
0.40	0.01022
0.50	0.01627
0.60	0.02653
0.70	0.04644
0.80	0.07079
0.90	0.16470
1.00	1.34373







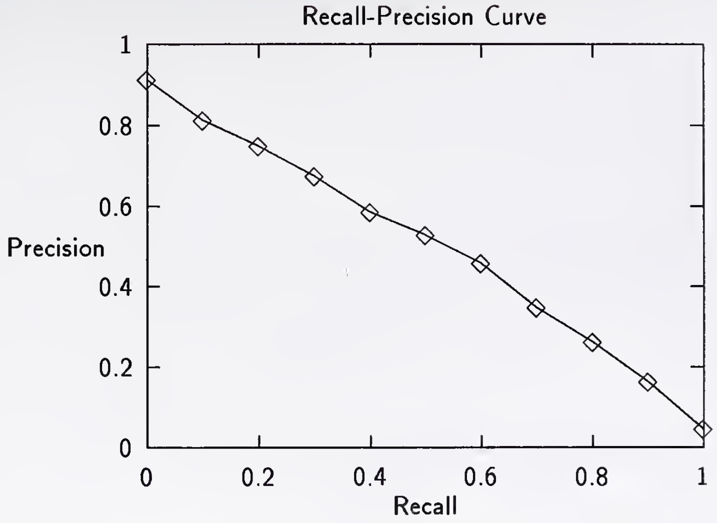
Summary Statistics	
Run Number	UCFSP1–category A, manual
Number of Topics	25

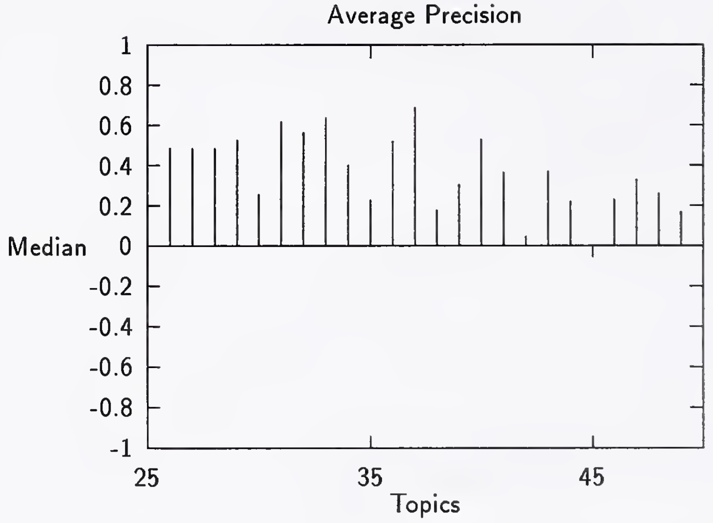
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2202
Rel_ret:	1898

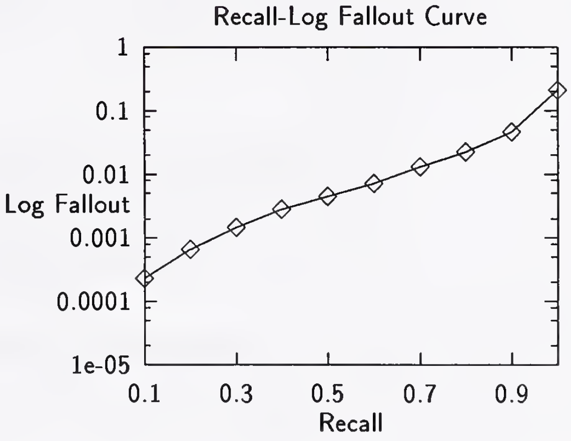
Recall Level Precision Averages	
Recall	Precision
0.00	0.9124
0.10	0.8118
0.20	0.7496
0.30	0.6742
0.40	0.5855
0.50	0.5275
0.60	0.4585
0.70	0.3483
0.80	0.2620
0.90	0.1636
1.00	0.0460
Average precision over all relevant docs	
non-interpolated	0.4918

Document Level Averages	
	Precision
At 5 docs	0.7680
At 10 docs	0.7600
At 15 docs	0.7413
At 20 docs	0.7100
At 30 docs	0.6773
At 100 docs	0.4960
At 200 docs	0.2880
At 500 docs	0.1426
At 1000 docs	0.0759
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.5041

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00023
0.20	0.00067
0.30	0.00145
0.40	0.00284
0.50	0.00449
0.60	0.00710
0.70	0.01312
0.80	0.02257
0.90	0.04609
1.00	0.20776





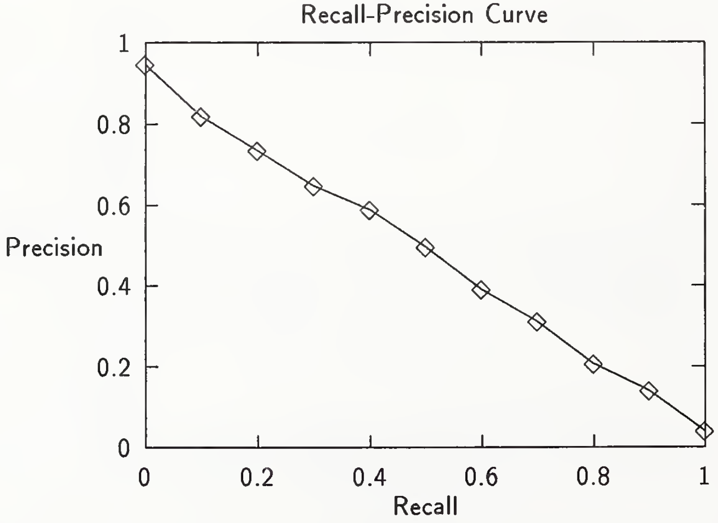


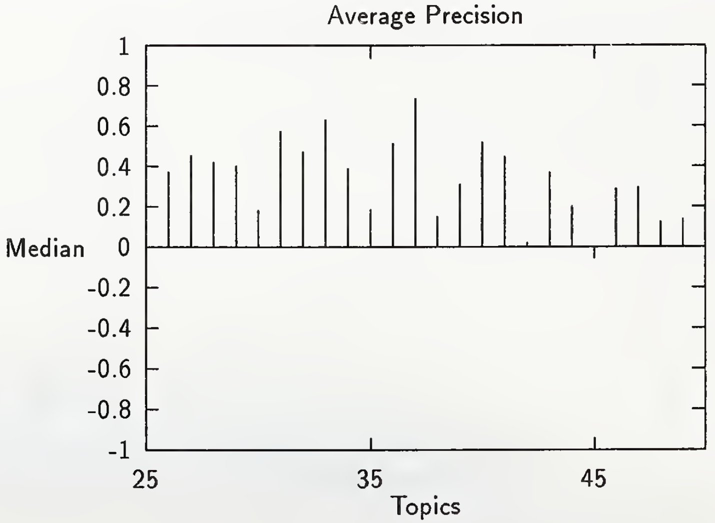
Summary Statistics	
Run Number	UCFSP2–category A, manual
Number of Topics	25
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2202
Rel_ret:	1843

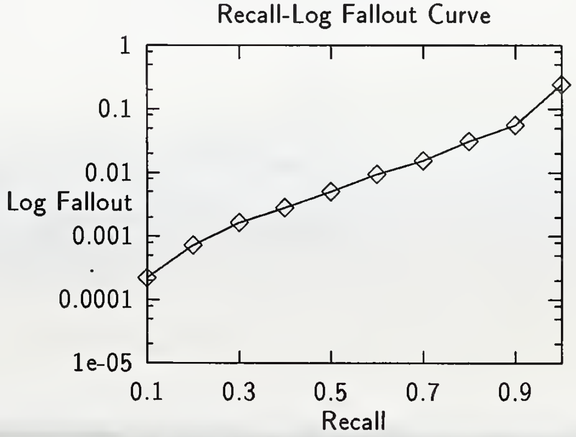
Recall Level Precision Averages	
Recall	Precision
0.00	0.9455
0.10	0.8181
0.20	0.7348
0.30	0.6476
0.40	0.5879
0.50	0.4963
0.60	0.3913
0.70	0.3118
0.80	0.2069
0.90	0.1403
1.00	0.0400
Average precision over all relevant docs	
non-interpolated	0.4695

Document Level Averages	
	Precision
At 5 docs	0.8000
At 10 docs	0.7720
At 15 docs	0.7413
At 20 docs	0.6980
At 30 docs	0.6587
At 100 docs	0.4644
At 200 docs	0.2766
At 500 docs	0.1348
At 1000 docs	0.0737
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.4874

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00022
0.20	0.00072
0.30	0.00164
0.40	0.00281
0.50	0.00508
0.60	0.00935
0.70	0.01548
0.80	0.03072
0.90	0.05525
1.00	0.24043





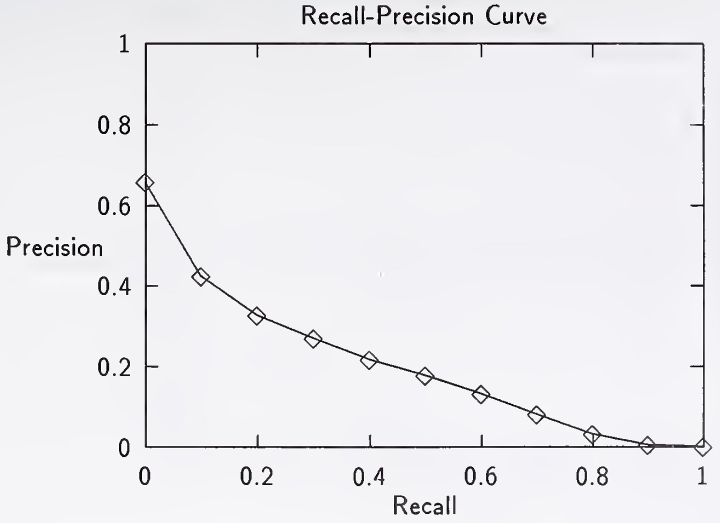


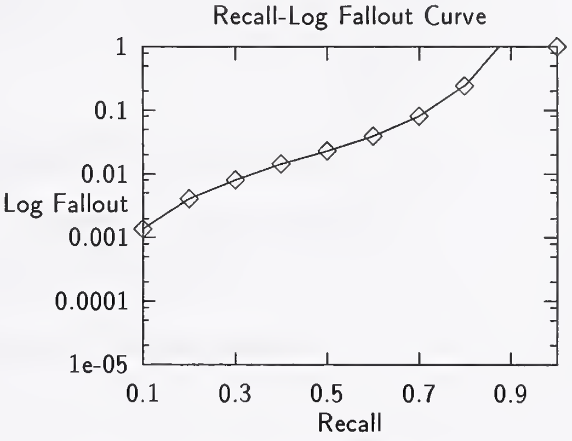
Summary Statistics	
Run Number	xerox-sp1–category A, automatic
Number of Topics	25
Total number of documents over all topics	
Retrieved:	25000
Relevant:	2202
Rel_ret:	1552

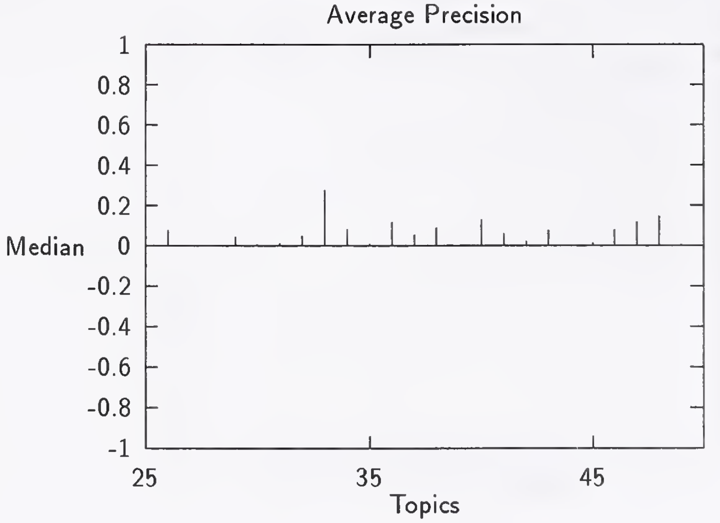
Recall Level Precision Averages	
Recall	Precision
0.00	0.6582
0.10	0.4247
0.20	0.3272
0.30	0.2705
0.40	0.2179
0.50	0.1788
0.60	0.1321
0.70	0.0806
0.80	0.0318
0.90	0.0053
1.00	0.0000
Average precision over all relevant docs	
non-interpolated	0.1899

Document Level Averages	
	Precision
At 5 docs	0.4320
At 10 docs	0.4120
At 15 docs	0.3840
At 20 docs	0.3660
At 30 docs	0.3440
At 100 docs	0.2448
At 200 docs	0.1770
At 500 docs	0.1017
At 1000 docs	0.0621
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2518

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00136
0.20	0.00412
0.30	0.00810
0.40	0.01438
0.50	0.02300
0.60	0.03949
0.70	0.07999
0.80	0.24400
0.90	1.69211
1.00	1.00000







Summary Statistics	
Run Number	xerox-sp2–category A, automatic
Number of Topics	25

Total number of documents over all topics	
Retrieved:	25000
Relevant:	2202
Rel_ret:	1728

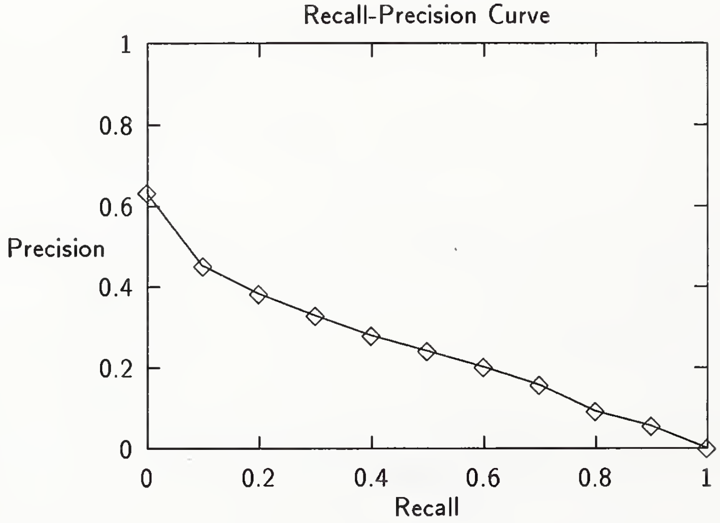
Recall Level Precision Averages	
Recall	Precision
0.00	0.6333
0.10	0.4514
0.20	0.3832
0.30	0.3295
0.40	0.2799
0.50	0.2410
0.60	0.2019
0.70	0.1573
0.80	0.0927
0.90	0.0553
1.00	0.0000

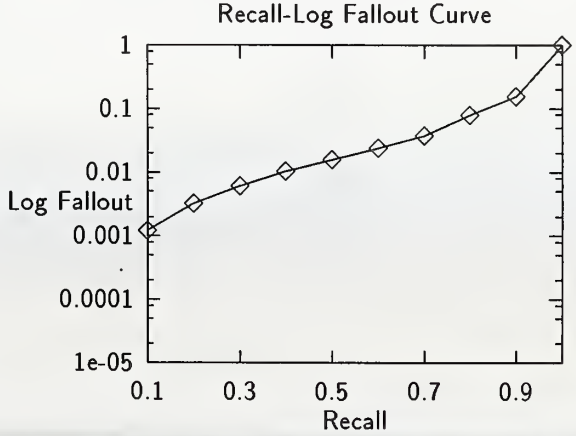
Average precision over all relevant docs	
non-interpolated	0.2378

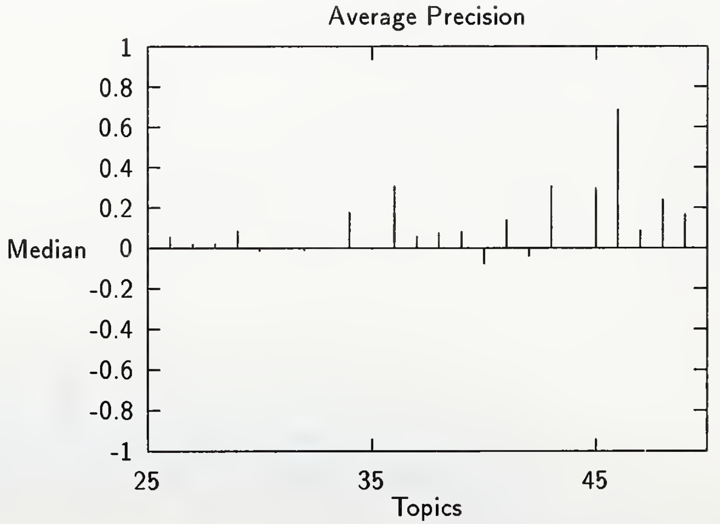
Document Level Averages	
	Precision
At 5 docs	0.4320
At 10 docs	0.4120
At 15 docs	0.3893
At 20 docs	0.3800
At 30 docs	0.3453
At 100 docs	0.2856
At 200 docs	0.1964
At 500 docs	0.1114
At 1000 docs	0.0691

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2840

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00122
0.20	0.00322
0.30	0.00612
0.40	0.01031
0.50	0.01577
0.60	0.02376
0.70	0.03757
0.80	0.07844
0.90	0.15402
1.00	1.00000







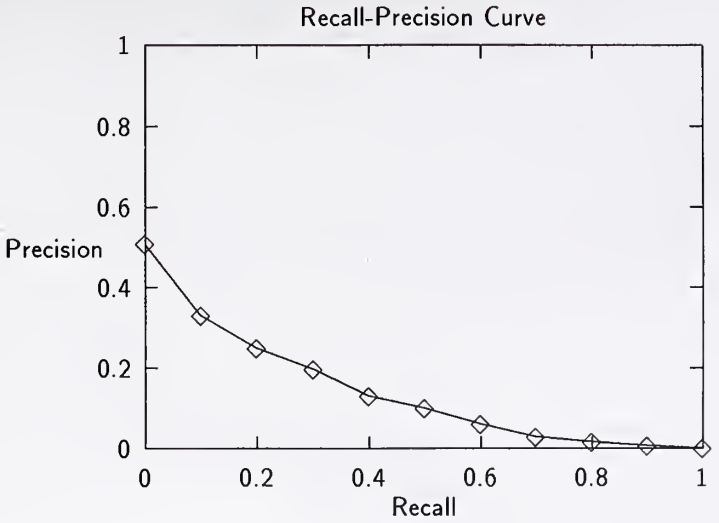
Summary Statistics	
Run Number	ACQC10–category B, manual
Number of Topics	49

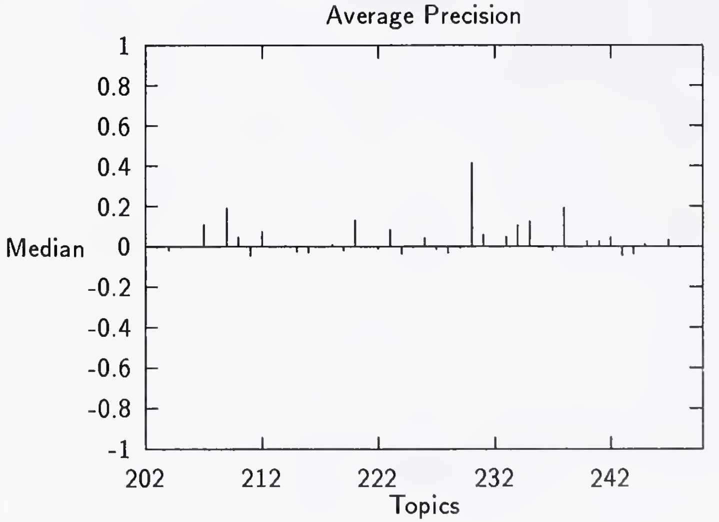
Total number of documents over all topics	
Retrieved:	48655
Relevant:	2480
Rel_ret:	1328

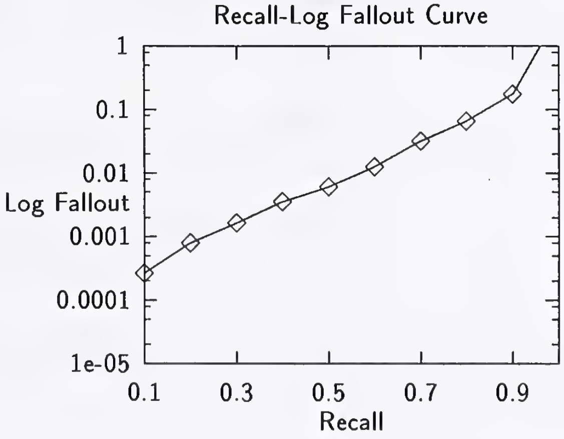
Recall Level Precision Averages	
Recall	Precision
0.00	0.5081
0.10	0.3307
0.20	0.2501
0.30	0.1975
0.40	0.1305
0.50	0.0996
0.60	0.0602
0.70	0.0283
0.80	0.0161
0.90	0.0069
1.00	0.0004
Average precision over all relevant docs	
non-interpolated	0.1297

Document Level Averages	
,	Precision
At 5 docs	0.2816
At 10 docs	0.2408
At 15 docs	0.2299
At 20 docs	0.2112
At 30 docs	0.1918
At 100 docs	0.1198
At 200 docs	0.0819
At 500 docs	0.0449
At 1000 docs	0.0271
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1681

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00027
0.20	0.00080
0.30	0.00163
0.40	0.00356
0.50	0.00603
0.60	0.01251
0.70	0.03209
0.80	0.06527
0.90	0.17294
1.00	3.33645







Summary Statistics	
Run Number	ACQC20–category B, manual
Number of Topics	49

Total number of documents over all topics	
Retrieved:	45951
Relevant:	2480
Rel_ret:	1143

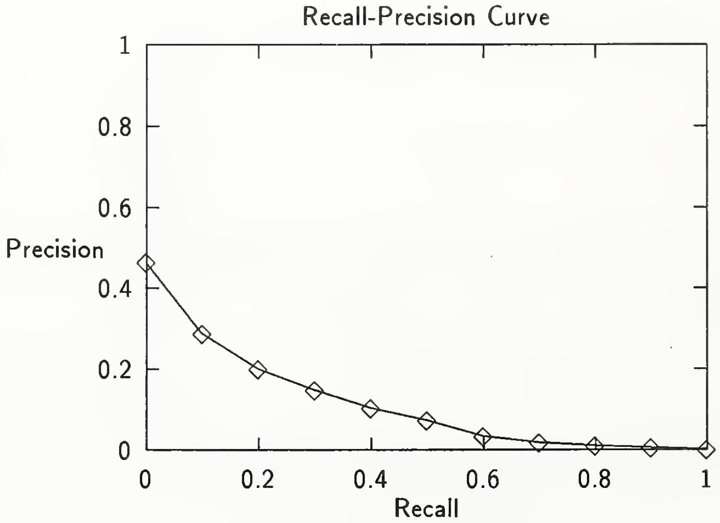
Recall Level Precision Averages	
Recall	Precision
0.00	0.4642
0.10	0.2881
0.20	0.2004
0.30	0.1475
0.40	0.1029
0.50	0.0730
0.60	0.0334
0.70	0.0172
0.80	0.0090
0.90	0.0048
1.00	0.0000

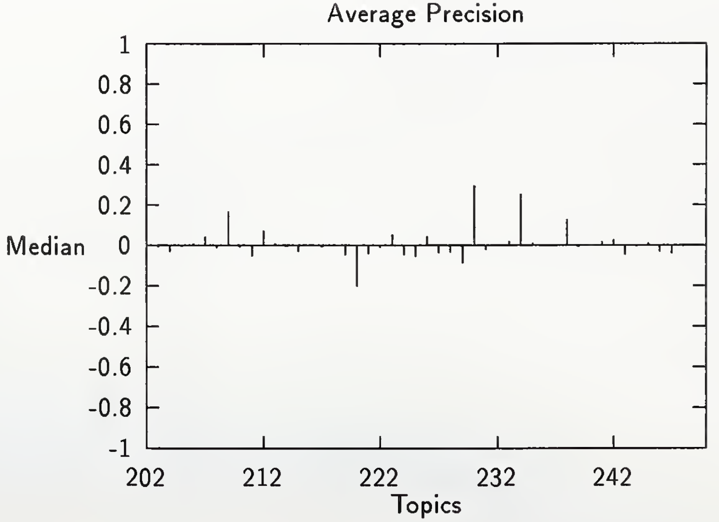
Average precision over all relevant docs	
non-interpolated	0.1044

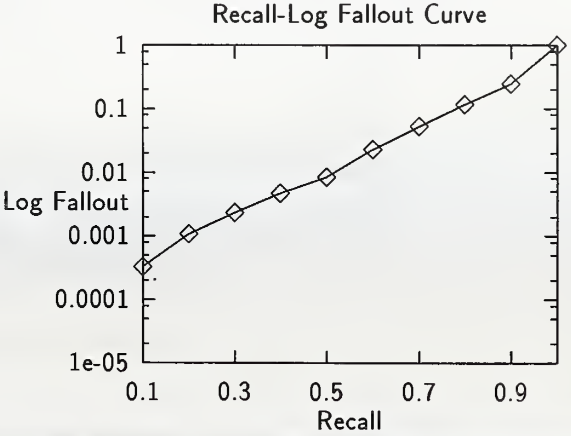
Document Level Averages	
	Precision
At 5 docs	0.2449
At 10 docs	0.2163
At 15 docs	0.2163
At 20 docs	0.1980
At 30 docs	0.1721
At 100 docs	0.1027
At 200 docs	0.0673
At 500 docs	0.0380
At 1000 docs	0.0233

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1508

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00033
0.20	0.00107
0.30	0.00231
0.40	0.00466
0.50	0.00848
0.60	0.02318
0.70	0.05340
0.80	0.11761
0.90	0.24913
1.00	1.00000







Summary Statistics	
Run Number	ACQUNC–category B, manual
Number of Topics	49

Total number of documents over all topics	
Retrieved:	48792
Relevant:	2480
Rel_ret:	1424

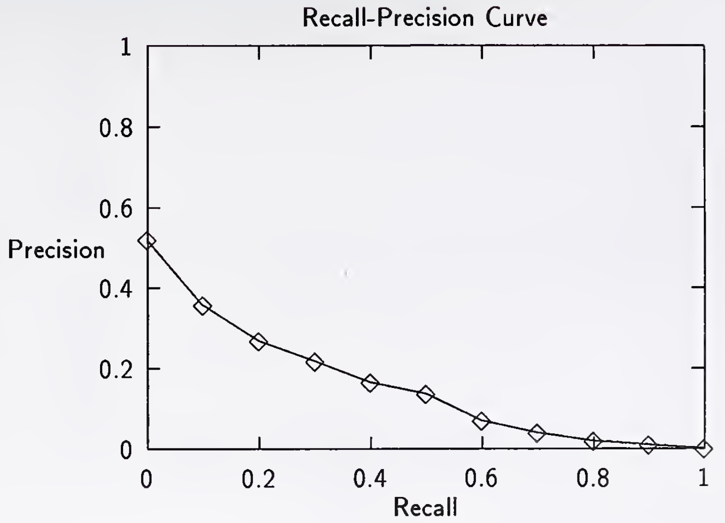
Recall Level Precision Averages	
Recall	Precision
0.00	0.5198
0.10	0.3576
0.20	0.2686
0.30	0.2171
0.40	0.1655
0.50	0.1373
0.60	0.0697
0.70	0.0396
0.80	0.0186
0.90	0.0088
1.00	0.0002

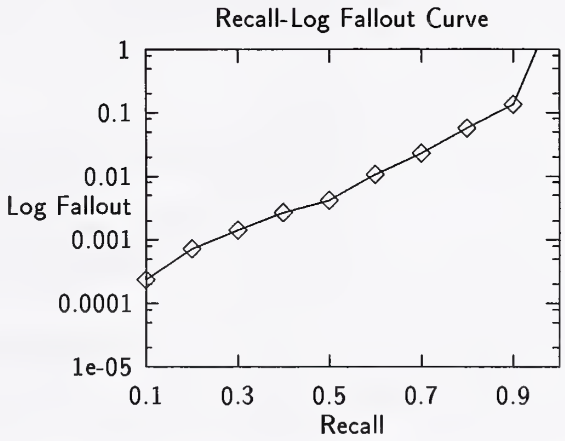
Average precision over all relevant docs	
non-interpolated	0.1454

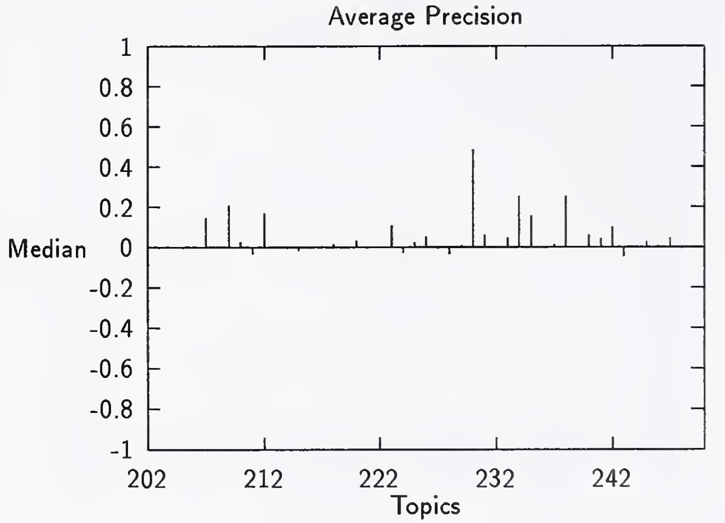
Document Level Averages	
	Precision
At 5 docs	0.2857
At 10 docs	0.2490
At 15 docs	0.2463
At 20 docs	0.2378
At 30 docs	0.2122
At 100 docs	0.1296
At 200 docs	0.0883
At 500 docs	0.0489
At 1000 docs	0.0291

R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1839

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00024
0.20	0.00073
0.30	0.00144
0.40	0.00269
0.50	0.00419
0.60	0.01069
0.70	0.02267
0.80	0.05636
0.90	0.13534
1.00	6.67423





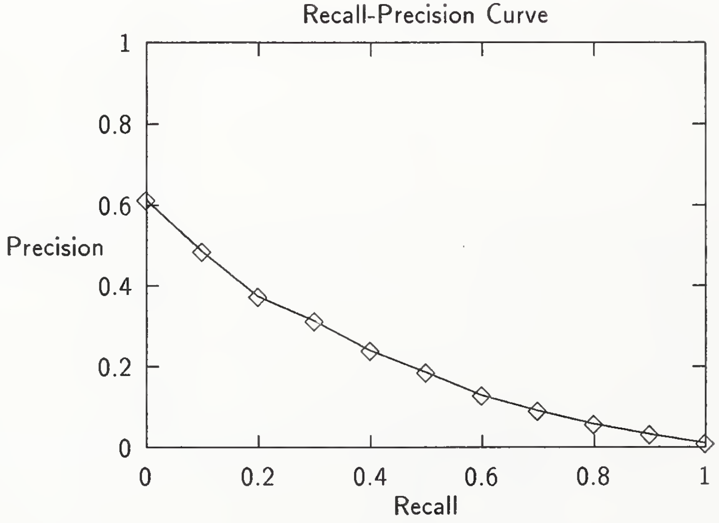


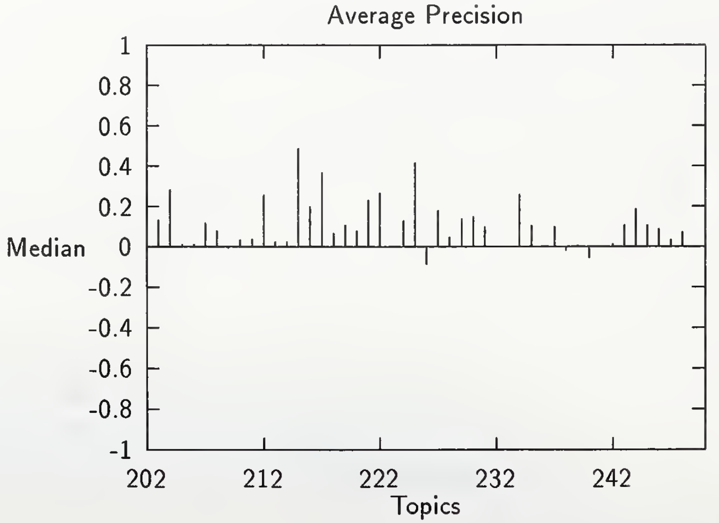
Summary Statistics	
Run Number	CrnlB–category B, automatic
Number of Topics	49
Total number of documents over all topics	
Retrieved:	49000
Relevant:	2480
Rel_ret:	1815

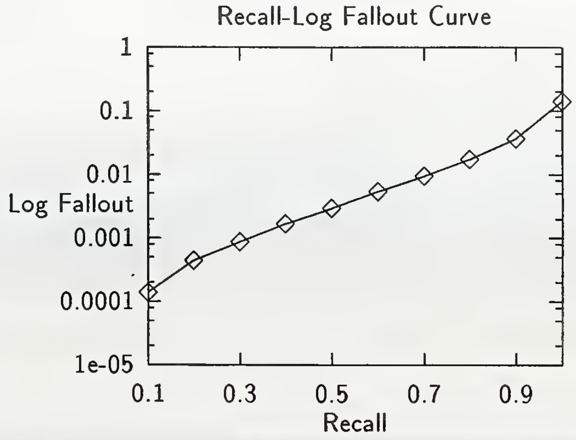
Recall Level Precision Averages	
Recall	Precision
0.00	0.6120
0.10	0.4848
0.20	0.3741
0.30	0.3127
0.40	0.2406
0.50	0.1860
0.60	0.1281
0.70	0.0908
0.80	0.0579
0.90	0.0320
1.00	0.0094
Average precision over all relevant docs	
non-interpolated	0.2084

Document Level Averages	
	Precision
At 5 docs	0.3796
At 10 docs	0.3408
At 15 docs	0.3156
At 20 docs	0.2980
At 30 docs	0.2816
At 100 docs	0.1673
At 200 docs	0.1160
At 500 docs	0.0634
At 1000 docs	0.0370
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2415

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00014
0.20	0.00045
0.30	0.00088
0.40	0.00169
0.50	0.00292
0.60	0.00545
0.70	0.00936
0.80	0.01738
0.90	0.03635
1.00	0.14070







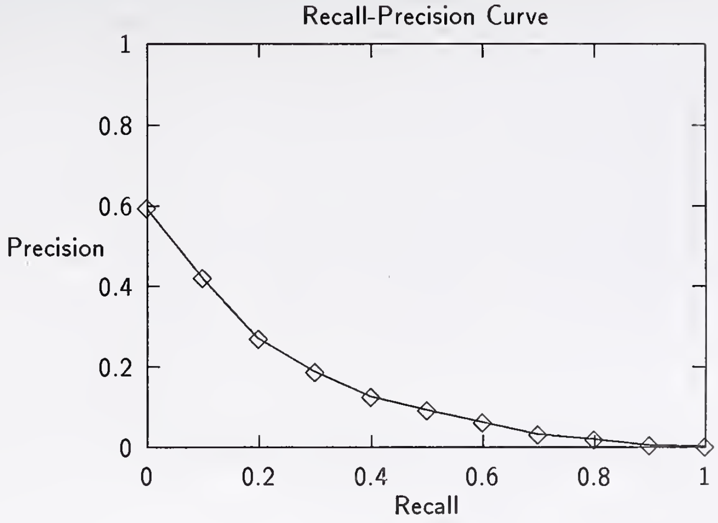
Summary Statistics	
Run Number	CrnlBc10–category B, automatic
Number of Topics	49

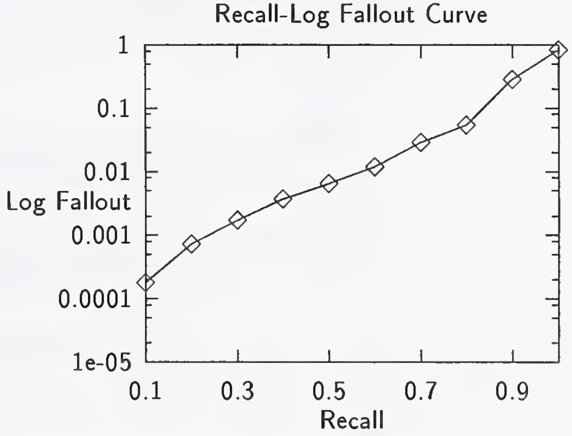
Total number of documents over all topics	
Retrieved:	49000
Relevant:	2480
Rel_ret:	1489

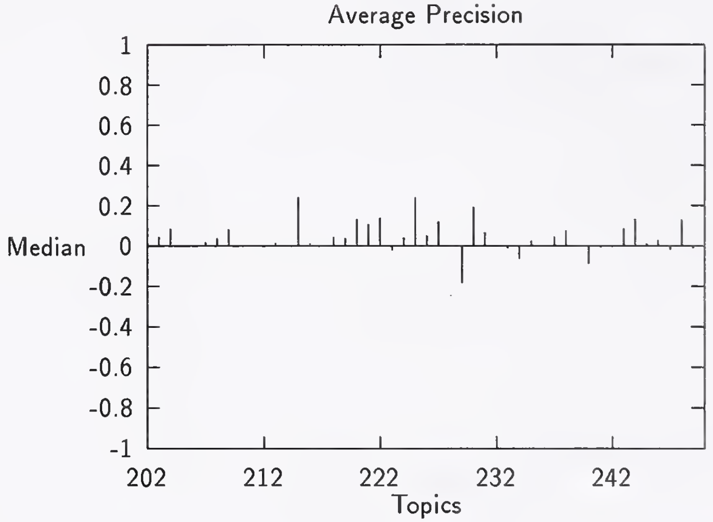
Recall Level Precision Averages	
Recall	Precision
0.00	0.5950
0.10	0.4213
0.20	0.2696
0.30	0.1877
0.40	0.1260
0.50	0.0924
0.60	0.0626
0.70	0.0316
0.80	0.0195
0.90	0.0042
1.00	0.0016
Average precision over all relevant docs	
non-interpolated	0.1431

Document Level Averages	
	Precision
At 5 docs	0.3469
At 10 docs	0.3102
At 15 docs	0.2830
At 20 docs	0.2592
At 30 docs	0.2231
At 100 docs	0.1302
At 200 docs	0.0923
At 500 docs	0.0511
At 1000 docs	0.0304
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1769

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00018
0.20	0.00072
0.30	0.00173
0.40	0.00370
0.50	0.00656
0.60	0.01200
0.70	0.02864
0.80	0.05371
0.90	0.28489
1.00	0.83311





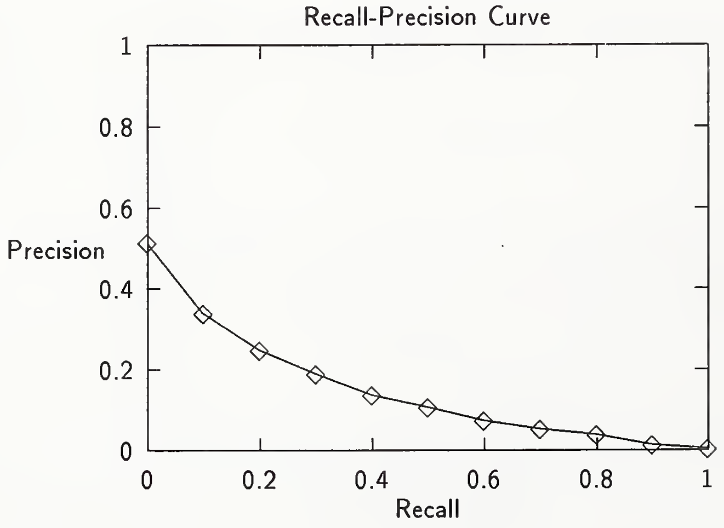


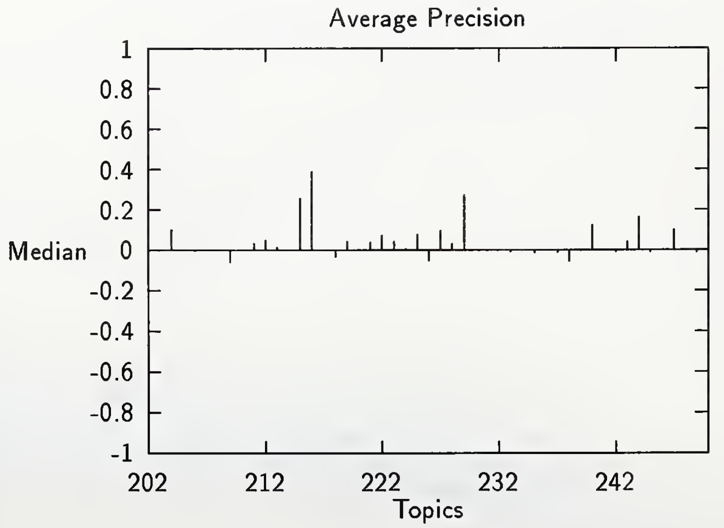
Summary Statistics	
Run Number	gmuc0–category B, automatic
Number of Topics	49
Total number of documents over all topics	
Retrieved:	49000
Relevant:	2480
Rel_ret:	1560

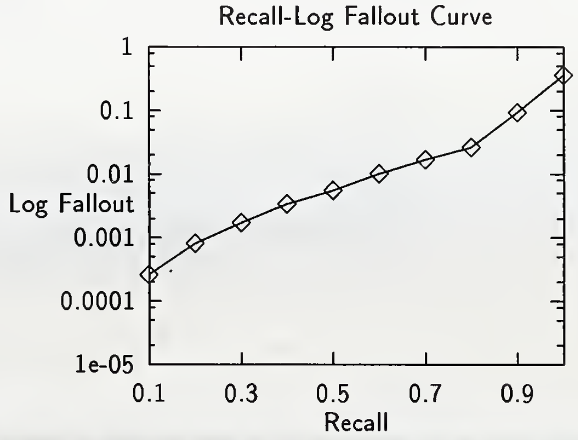
Recall Level Precision Averages	
Recall	Precision
0.00	0.5132
0.10	0.3384
0.20	0.2487
0.30	0.1898
0.40	0.1362
0.50	0.1068
0.60	0.0735
0.70	0.0527
0.80	0.0386
0.90	0.0128
1.00	0.0037
Average precision over all relevant docs	
non-interpolated	0.1401

Document Level Averages	
	Precision
At 5 docs	0.2898
At 10 docs	0.2735
At 15 docs	0.2435
At 20 docs	0.2245
At 30 docs	0.1925
At 100 docs	0.1235
At 200 docs	0.0894
At 500 docs	0.0514
At 1000 docs	0.0318
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1673

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00026
0.20	0.00081
0.30	0.00171
0.40	0.00339
0.50	0.00558
0.60	0.01010
0.70	0.01680
0.80	0.02660
0.90	0.09267
1.00	0.35951





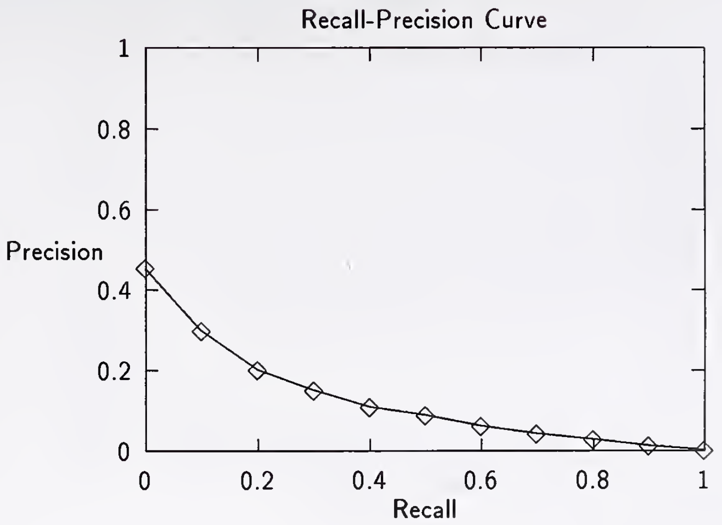


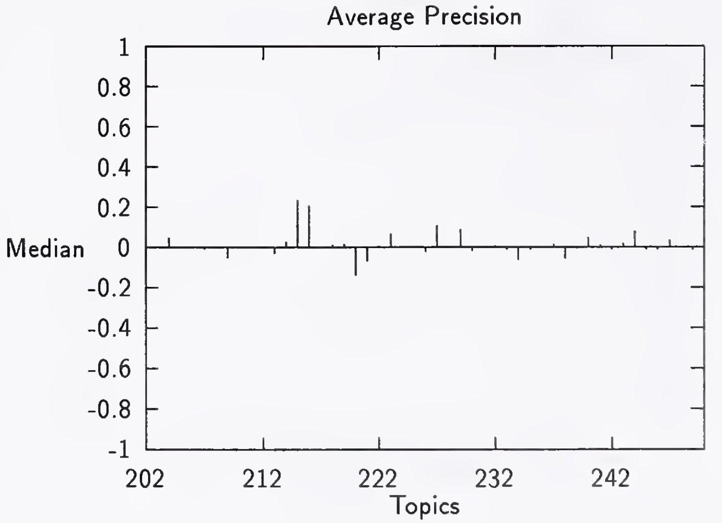
Summary Statistics	
Run Number	gmuc10–category B, automatic
Number of Topics	49
Total number of documents over all topics	
Retrieved:	49000
Relevant:	2480
Rel_ret:	1521

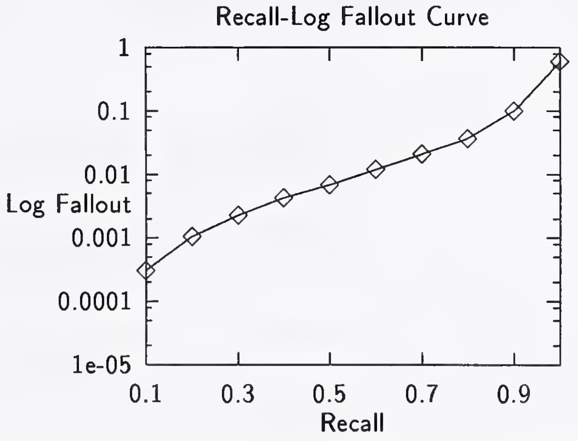
Recall Level Precision Averages	
Recall	Precision
0.00	0.4544
0.10	0.2992
0.20	0.2013
0.30	0.1503
0.40	0.1104
0.50	0.0883
0.60	0.0627
0.70	0.0431
0.80	0.0286
0.90	0.0120
1.00	0.0022
Average precision over all relevant docs	
non-interpolated	0.1153

Document Level Averages	
	Precision
At 5 docs	0.3061
At 10 docs	0.2347
At 15 docs	0.2054
At 20 docs	0.1898
At 30 docs	0.1639
At 100 docs	0.1129
At 200 docs	0.0824
At 500 docs	0.0490
At 1000 docs	0.0310
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.1449

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00031
0.20	0.00106
0.30	0.00226
0.40	0.00430
0.50	0.00689
0.60	0.01198
0.70	0.02075
0.80	0.03628
0.90	0.09893
1.00	0.60553





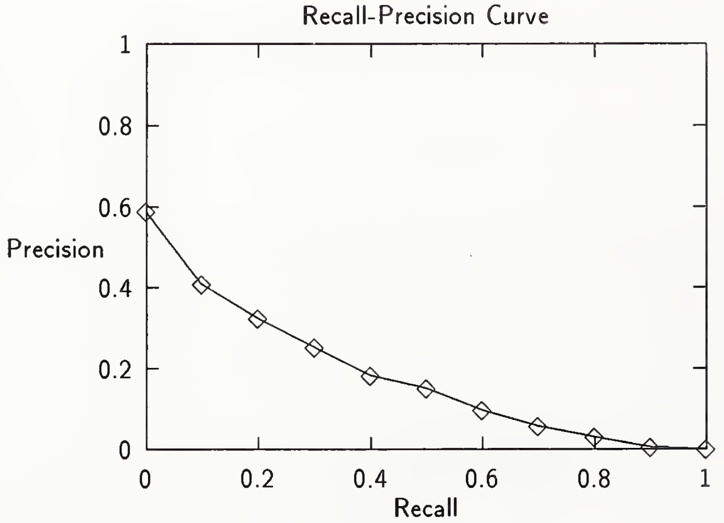


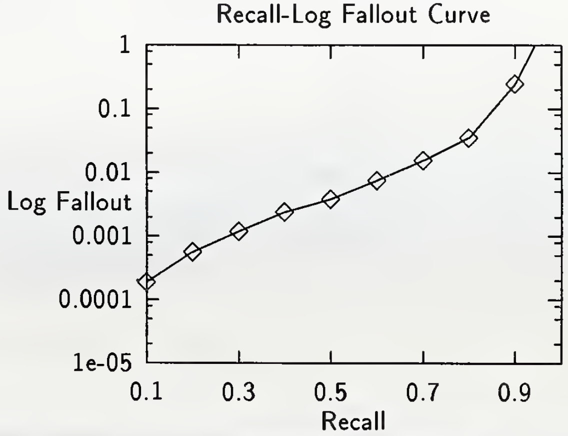
Summary Statistics	
Run Number	rutfum–category B, manual
Number of Topics	49
Total number of documents over all topics	
Retrieved:	49000
Relevant:	2480
Rel_ret:	1611

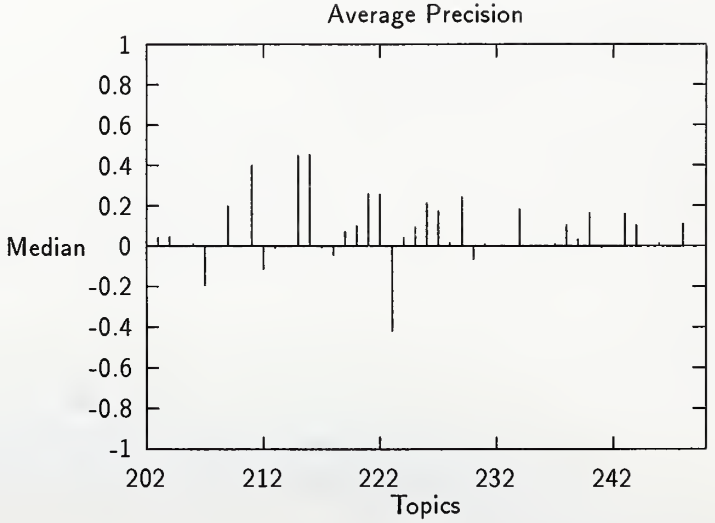
Recall Level Precision Averages	
Recall	Precision
0.00	0.5882
0.10	0.4092
0.20	0.3248
0.30	0.2532
0.40	0.1832
0.50	0.1503
0.60	0.0973
0.70	0.0569
0.80	0.0295
0.90	0.0048
1.00	0.0002
Average precision over all relevant docs	
non-interpolated	0.1718

Document Level Averages	
	Precision
At 5 docs	0.3673
At 10 docs	0.3224
At 15 docs	0.2762
At 20 docs	0.2622
At 30 docs	0.2361
At 100 docs	0.1449
At 200 docs	0.0976
At 500 docs	0.0532
At 1000 docs	0.0329
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2033

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00019
0.20	0.00056
0.30	0.00118
0.40	0.00238
0.50	0.00377
0.60	0.00743
0.70	0.01549
0.80	0.03514
0.90	0.24913
1.00	6.67423







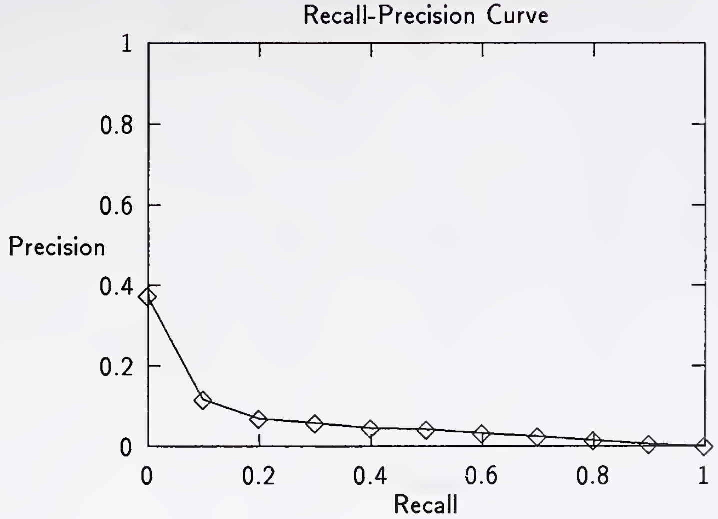
Summary Statistics	
Run Number	rutfuv–category B, manual
Number of Topics	49

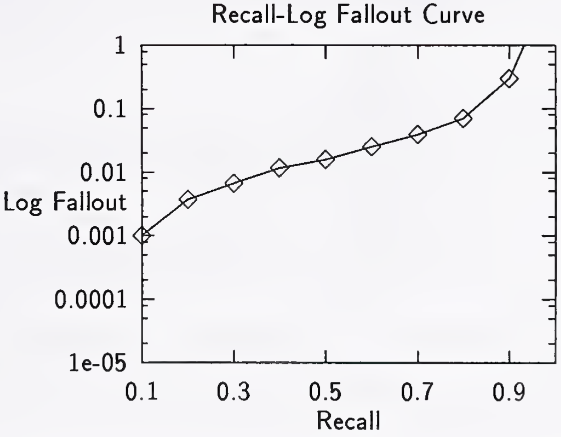
Total number of documents over all topics	
Retrieved:	49000
Relevant:	2480
Rel_ret:	1543

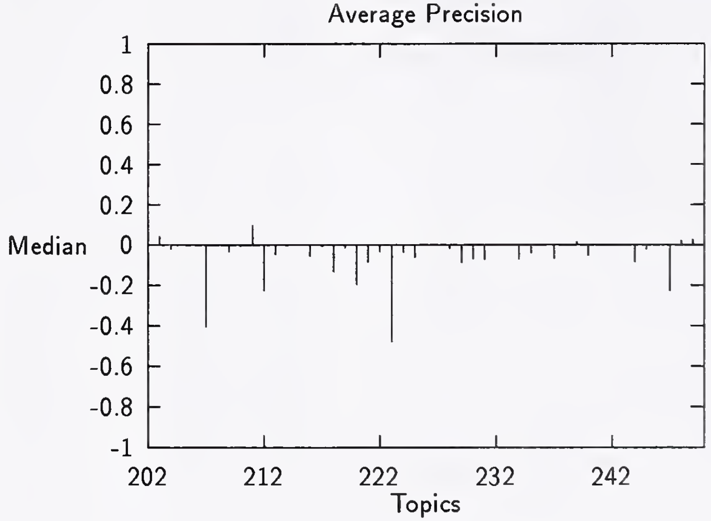
Recall Level Precision Averages	
Recall	Precision
0.00	0.3743
0.10	0.1161
0.20	0.0677
0.30	0.0573
0.40	0.0441
0.50	0.0407
0.60	0.0311
0.70	0.0234
0.80	0.0150
0.90	0.0040
1.00	0.0001
Average precision over all relevant docs	
non-interpolated	0.0484

Document Level Averages	
	Precision
At 5 docs	0.1306
At 10 docs	0.0878
At 15 docs	0.0694
At 20 docs	0.0612
At 30 docs	0.0537
At 100 docs	0.0508
At 200 docs	0.0409
At 500 docs	0.0365
At 1000 docs	0.0315
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0643

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00102
0.20	0.00368
0.30	0.00659
0.40	0.01158
0.50	0.01573
0.60	0.02496
0.70	0.03900
0.80	0.07014
0.90	0.29920
1.00	13.34980





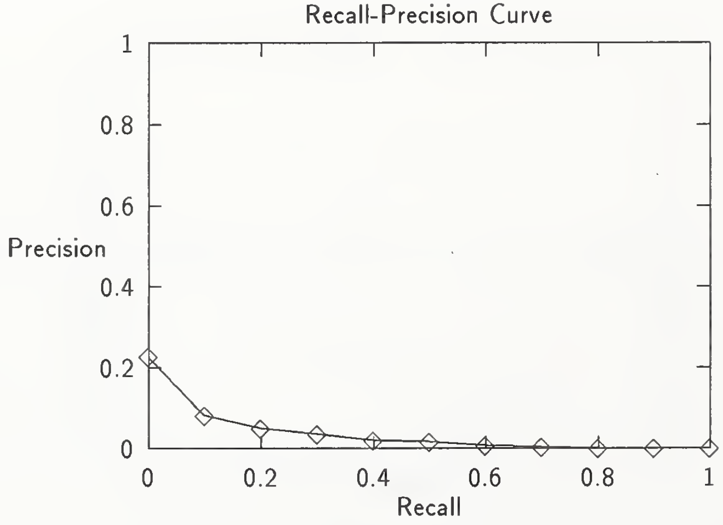


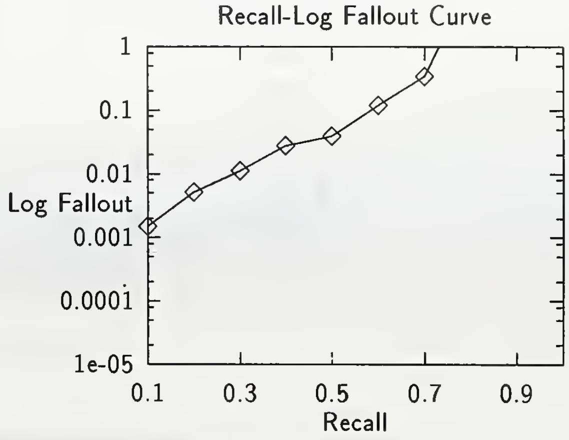
Summary Statistics	
Run Number	rutscn20–category B, manual
Number of Topics	49
Total number of documents over all topics	
Retrieved:	49000
Relevant:	2480
Rel_ret:	452

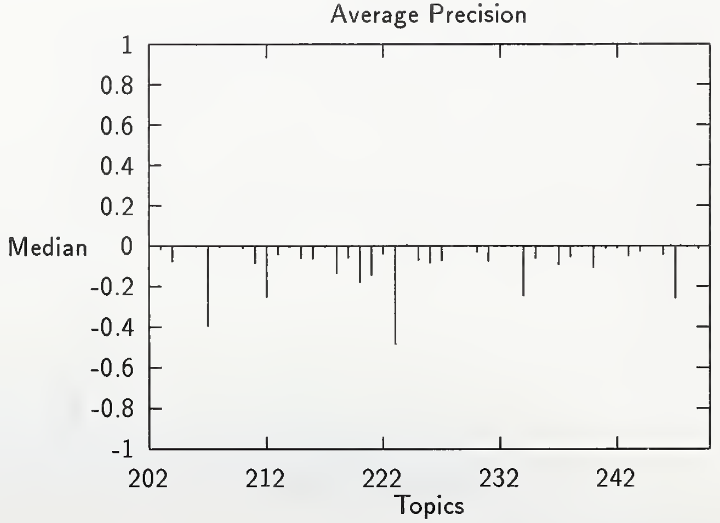
Recall Level Precision Averages	
Recall	Precision
0.00	0.2278
0.10	0.0811
0.20	0.0493
0.30	0.0345
0.40	0.0190
0.50	0.0166
0.60	0.0067
0.70	0.0027
0.80	0.0001
0.90	0.0001
1.00	0.0001
Average precision over all relevant docs	
non-interpolated	0.0268

Document Level Averages	
	Precision
At 5 docs	0.1061
At 10 docs	0.0980
At 15 docs	0.0830
At 20 docs	0.0755
At 30 docs	0.0667
At 100 docs	0.0416
At 200 docs	0.0291
At 500 docs	0.0162
At 1000 docs	0.0092
R–Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.0615

Recall Fallout Averages	
Recall	Fallout
0.00	0.00000
0.10	0.00151
0.20	0.00515
0.30	0.01121
0.40	0.02757
0.50	0.03955
0.60	0.11876
0.70	0.34521
0.80	10.67984
0.90	12.01482
1.00	13.34980







FILTERING RESULTS

This appendix contains results for all TREC-4 filtering participants.

Evaluation Techniques and Measures

Filtering Task

This track tests a methodology for evaluating binary text classification systems, i.e. IR systems which accept or reject a document as it is processed. In order to test customization of systems to varying user preferences, three separate runs are submitted by each system: Run 1 is for high precision, Run 2 is for medium precision, and Run 3 is for low precision.

System Results Description

The results for each system consist of three pages of results: the first page contains a table of the raw data, the second and third pages include three tables and six graphs derived from the raw data to evaluate the performance in various categories and from different perspectives. A final set of 6 multi-site graphs compare all systems.

Tables

Table 1 is generated by programs specifically developed for the filtering task using methodology as defined in Lewis[1]. Tables 2 and 3 are created from the traditional adhoc and routing types of data. Table 4 is specific to the filtering track and is a comparison of all systems in the filtering track.

- Table 1 is a "Raw Data" Table for the three runs from a given system.

TABLE 1

Raw Data															
	Run 1					Run 2					Run 3				
topic	set	pool	sample			set	pool	sample			set	pool	sample		
	size	util.	util.			size	util.	util.			size	util.	util.		
3	4	0.0	0.0	+/-	0.0	24	18.0	18.0	+/-	0.0	47	105.0	105.0	+/-	0.0
5	26	26.0	26.0	+/-	0.0	59	41.0	41.0	+/-	0.0	106	146.0	149.0	+/-	9.5
14	19	-57.0	-57.0	+/-	0.0	19	-19.0	-19.0	+/-	0.0	28	-20.0	-20.0	+/-	0.0
17	0	0.0	0.0	+/-	0.0	33	23.0	23.0	+/-	0.0	269	535.0	575.7	+/-	80.3
20	79	11.0	8.8	+/-	32.1	133	71.0	70.1	+/-	37.2	162	334.0	332.2	+/-	37.2
27	3	-9.0	-9.0	+/-	0.0	68	-22.0	-24.7	+/-	17.9	246	-6.0	44.1	+/-	79.5
28	14	10.0	10.0	+/-	0.0	44	26.0	26.0	+/-	0.0	118	214.0	217.6	+/-	18.3
29	3	-1.0	-1.0	+/-	0.0	6	4.0	4.0	+/-	0.0	28	24.0	24.0	+/-	0.0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

All subsequent graphs and tables are based on this data.

- Run

  The three runs correspond to the three user preferences: Run 1 for high precision, Run 2 for medium precision, and Run 3 for low precision. The utility measure awards positive points to relevant documents and negative points to nonrelevant documents. Let A_i be the number of relevant documents in the submitted set for Run i, denoted R_i; and let B_i be the number of nonrelevant documents in R_i. Utility measures can be defined corresponding to the user preferences. The utilities used to measure the three runs are defined as follows:

  $$\begin{array}{ll} \text{Run 1} & u_1 = A_1 - 3B_1 \\ \text{Run 2} & u_2 = A_2 - B_2 \\ \text{Run 3} & u_3 = 3A_3 - B_3. \end{array}$$

  Run 1 awards fewer positive points for relevant documents than negative points for nonrelevant documents, simulating the users need for high precision or a set of documents where a marginally relevant document is not returned. Run 2 awards equal positive and negative points for relevant and nonrelevant documents. Run 3 is the reverse of Run 1 where more positive points are awarded for relevant documents than negative points for nonrelevant documents, simulating the users need for high recall or a set of documents where a marginally relevant document is returned.

- Set size

  The *set size* is the number of documents submitted for each run for that topic.

- Pool utility

  The *pool utility* column indicates the utility of each submitted set under the assumption that all relevant documents got judged as part of the normal routing pool.

- Sample utility

  The *sample utility* column is an estimate of the utility of each submitted set using stratified random sampling [1].

- Table 2 is a "Summary Statistics" Table.

TABLE 2

Summary Statistics			
Run Number	xerox–category A, manual		
Number of Topics			50
Total number of documents over all topics			
	Run 1	Run 2	Run 3
Retrieved:	941	2463	5735
Relevant:	6576	6576	6576
Rel_ret:	639	1405	2363

This table is the same table produced for adhoc and routing in Appendix A, only the filtering "Summary Statistics" Table contains three columns, one for each run.

- Table 3 is a "Average Effectiveness (Pooled Sample)" Table.

Table 3

Average Effectiveness (Pooled Sample)			
	Recall	Precision	Fallout
Run 1	0.095	0.527	0.0000
Run 2	0.190	0.577	0.0001
Run 3	0.312	0.487	0.0003

The recall, precision, and fallout are computed for each run. See Appendix A for a description of recall, precision, and fallout.

- Table 4 is a sample "Relative Ranks" Table.

Table 4

Relative Ranks Number of topics where this site's utility was					
	1st	2nd	3rd	4th	Mean Rank
		Pooled			
Run 1	21	16	9	4	1.959
Run 2	23	17	8	2	1.816
Run 3	18	21	9	2	1.939
		Sampled			
Run 1	21	15	10	4	1.980
Run 2	21	18	9	2	1.878
Run 3	18	23	7	2	1.898

The table contains data for both pooled and sampled utilities. Each row is a set of *relative ranks* for a particular run. TREC-4 had four systems participating, so there are four possible ranks. The mean rank is the mean average of the row except in the case of ties. Ties are computed by assigning the ranks of the tied systems and rounding the average down to the next best rank. (i.e. two systems tie for first so the ranks one and two are added and divided by the number of systems, two; 1+2/2 = 1.5; round(1.5) = 1. Both systems are given a rank of one.)

Utility Graphs

- Figure 1 is a "Estimated Sample Utility" Graph for Run 1 and Figure 2 is a "Estimated Pool Utility" Graph for Run 1.

The filtering results contain both types of utility graphs, one for each run. The y-axis is the estimated sample or pool utility ranging from -600 to 600, the minimum and maximum possible utility for an arbitrary run, The x-axis is the topic numbers sorted in order of best utility to worst utility. Each point on the system sample utility curve has a confidence

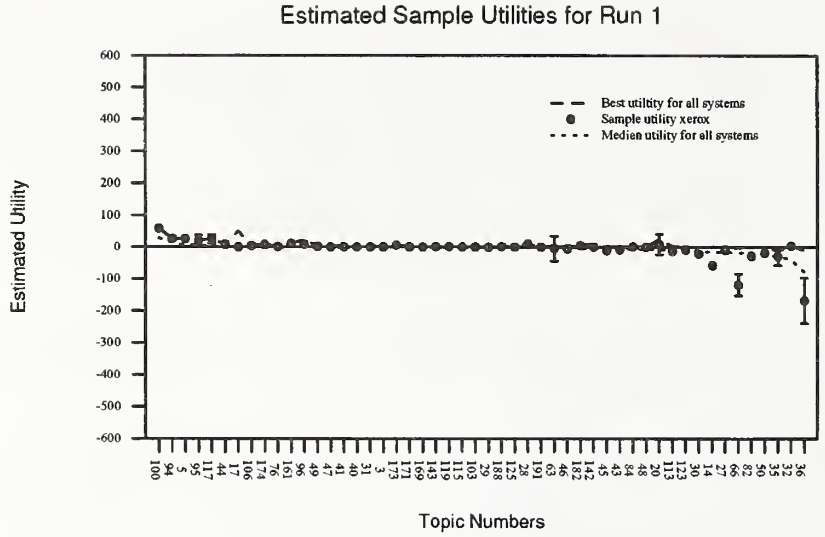


FIG. 1.

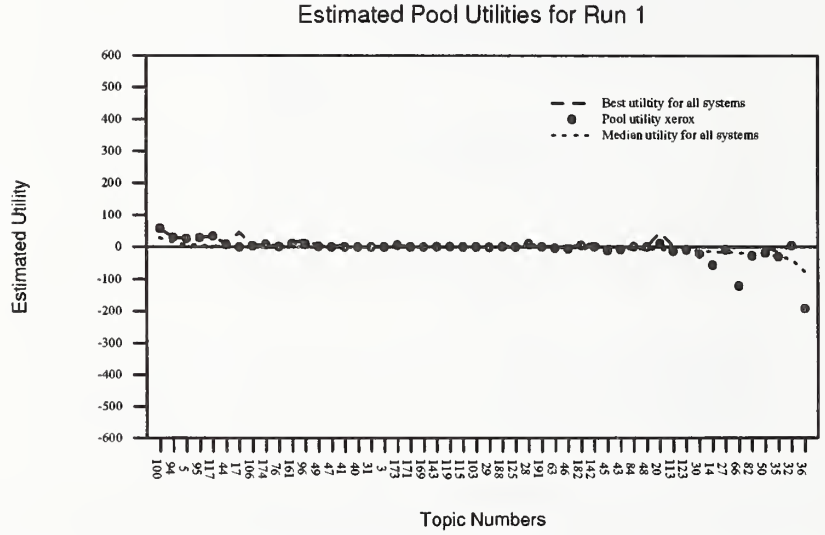


FIG. 2. *Estimated Pool Utility Graphs for Run 1.*

interval associated with it. If the confidence intervals are the point itself, no confidence bars are given. Each graph contains three curves:

I. The sample or pool utility for that system.

II. The median utility for all systems of that run.

III. The best utility for that run.

These graphs illustrate how a particular sample or pool utility run compares with an entire set of submitted systems. If the best utility curve and the median utility curve are close to each other the results of all systems in that run did not have much variation.

Multi-site comparisons

- Figures 3 and 4 are examples of multi-site graphs.

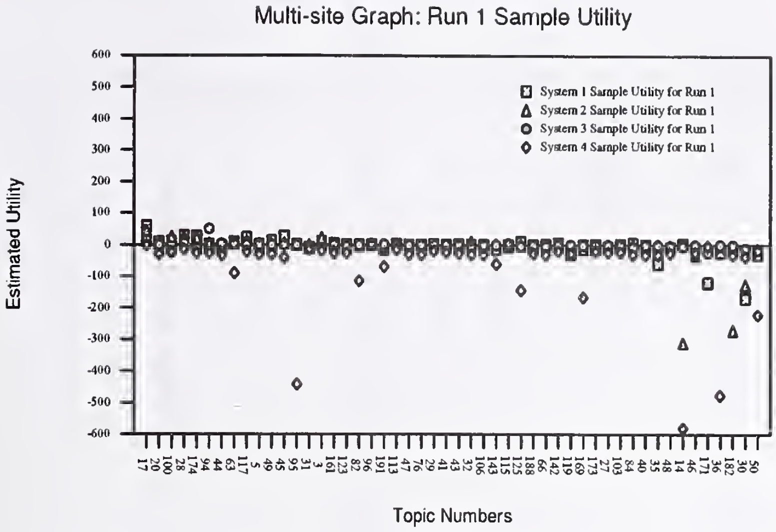


FIG. 3. *Sample utility for Run 1.*

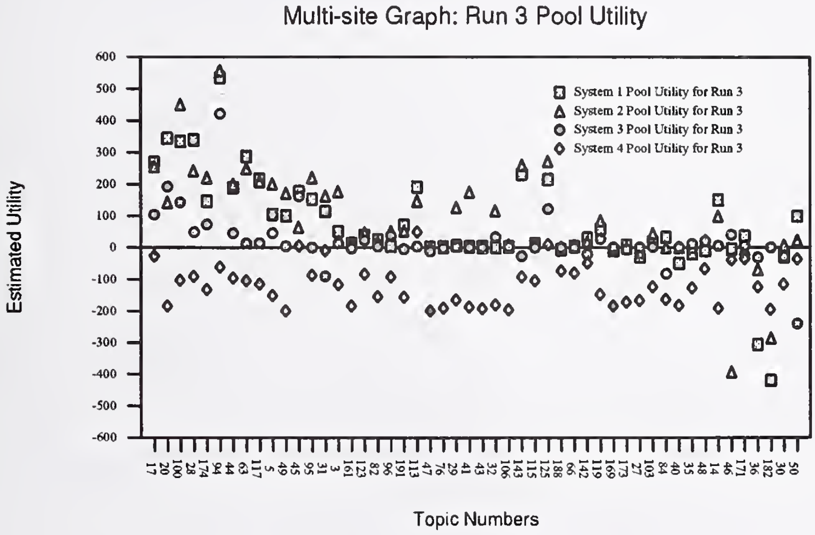


FIG. 4. *Multi-site Estimated Pool Utility Graphs for Run 3.*

There are six multi-site graphs, three for each run using sample utility and three for each run using pool utility. These are the same as figures 1 through 2 only the set of points are from each of the filtering systems. These graphs are only useful for seeing which system contributed outliers to the overall results of the filtering run. Figure 3 is typical of a high precision run with utilities around zero because fewer documents are returned. System 4 contributed the most negative outliers. Figure 4 is a high recall run with much more variation in results. System 1 and system 2 contributed more positive outliers while system 4 contributed negative outliers.

References

[1] D. Lewis, *The trec-4 filtering track*, in The Fourth Text REtrieval Conference (TREC-4), 1996.

Summary Statistics			
Run Number	ACQ–category A, manual		
Number of Topics			50
Total number of documents over all topics			
	Run 1	Run 2	Run 3
Retrieved:	1386	4107	9262
Relevant:	6576	6576	6576
Rel_ret:	230	571	856

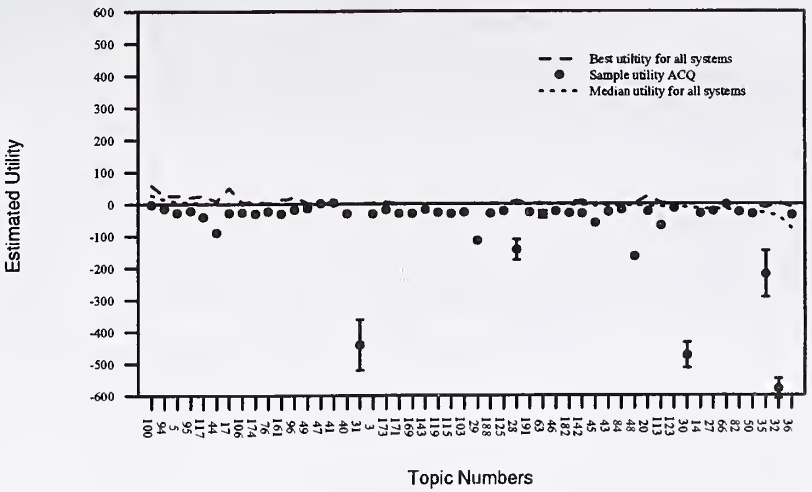
Average Effectiveness (Pooled Sample)

	Recall	Precision	Fallout
Run 1	0.032	0.205	0.0001
Run 2	0.078	0.135	0.0003
Run 3	0.138	0.091	0.0008

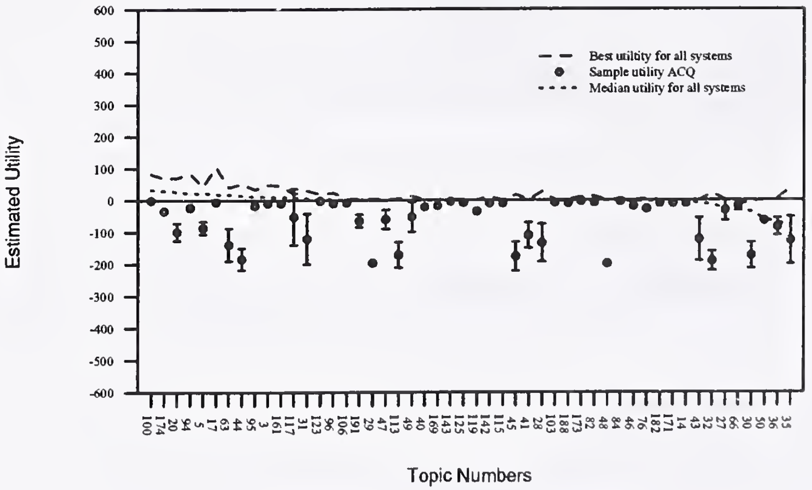
Number of topics where this site's Utility was

	1st	2nd	3rd	4th	Mean Rank
			Pooled		
Run 1	3	1	5	41	3.755
Run 2	0	1	9	40	3.857
Run 3	0	1	5	44	3.939
			Sampled		
Run 1	3	1	5	41	3.755
Run 2	0	1	10	39	3.837
Run 3	0	1	9	40	3.857

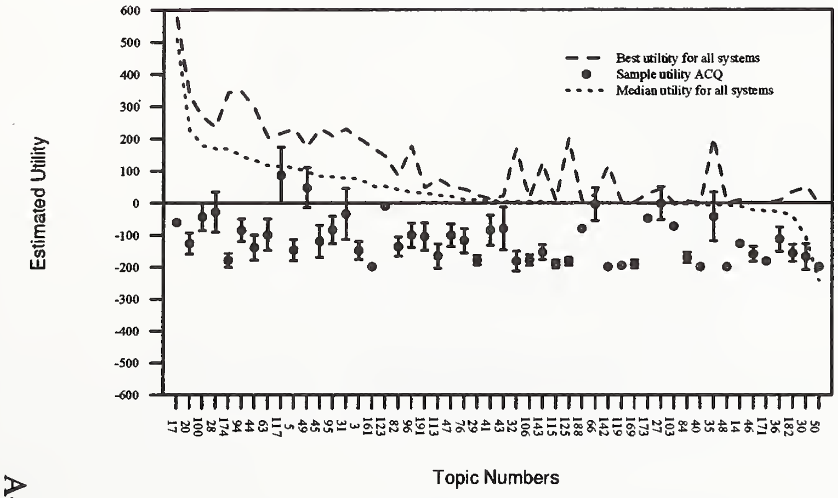
Estimated Sample Utilities for Run 1



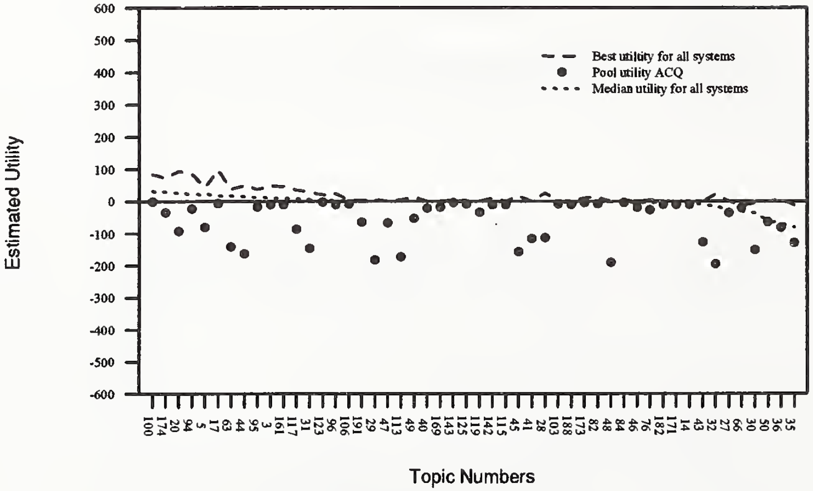
Estimated Sample Utilities for Run 2



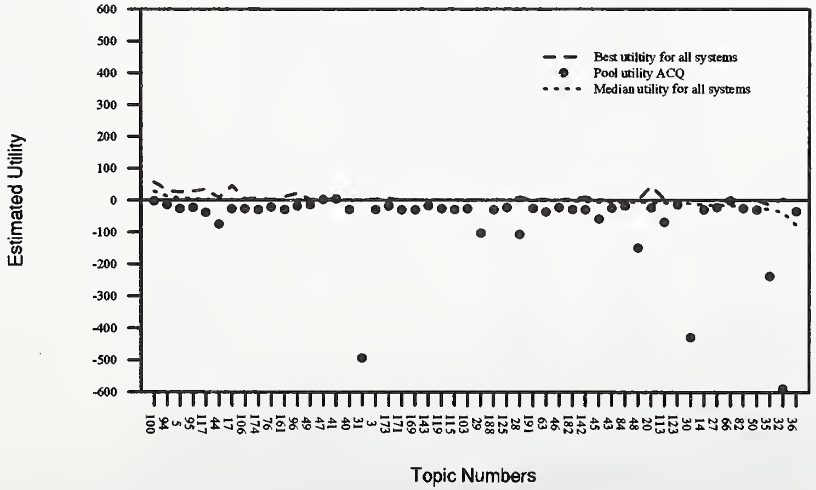
Estimated Sample Utilities for Run 3



Estimated Pool Utilities for Run 2



Estimated Pool Utilities for Run 1



Estimated Pool Utilities for Run 3

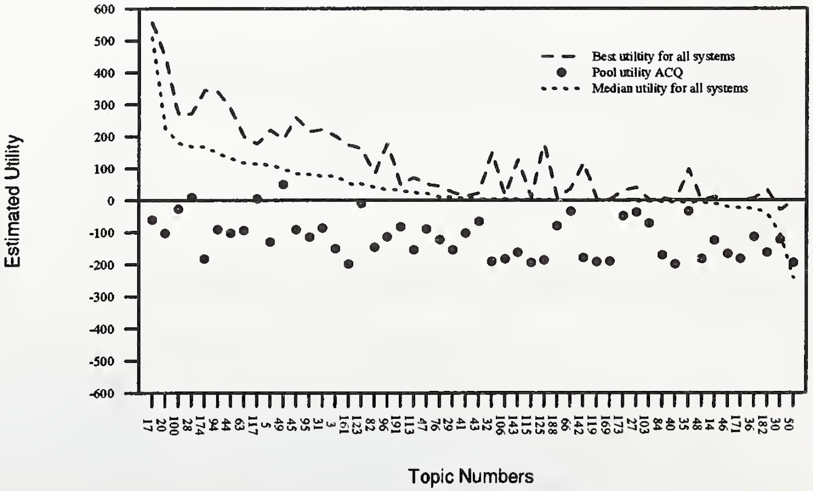


Table 1: Raw Data

topic	Run 1					Run 2					Run 3				
	set size	pool util.	sample util.			set size	pool util.	sample util.			set size	pool util.	sample util.		
3	10	-30.0	-30.0	+/-	0.0	10	-10.0	-10.0	+/-	0.0	199	-151.0	-148.3	+/-	28.6
5	10	-26.0	-26.0	+/-	0.0	103	-81.0	-86.5	+/-	19.8	199	-131.0	-146.7	+/-	32.6
14	10	-30.0	-30.0	+/-	0.0	10	-10.0	-10.0	+/-	0.0	126	-126.0	-126.0	+/-	0.0
17	10	-26.0	-26.0	+/-	0.0	10	-6.0	-6.0	+/-	0.0	97	-61.0	-61.0	+/-	0.0
20	16	-24.0	-24.0	+/-	0.0	127	-93.0	-99.0	+/-	27.9	199	-103.0	-126.2	+/-	32.5
27	17	-23.0	-23.0	+/-	0.0	99	-35.0	-31.6	+/-	31.8	199	-39.0	-0.8	+/-	51.6
28	65	-107.0	-143.3	+/-	32.2	199	-113.0	-133.8	+/-	58.9	199	9.0	-28.1	+/-	62.8
29	40	-104.0	-113.6	+/-	7.7	199	-183.0	-195.8	+/-	7.7	199	-155.0	-178.5	+/-	14.7
30	170	-430.0	-474.2	+/-	40.1	199	-151.0	-173.1	+/-	40.1	199	-123.0	-167.8	+/-	40.7
31	199	-493.0	-441.2	+/-	79.7	199	-147.0	-121.1	+/-	79.7	199	-87.0	-34.2	+/-	79.8
32	199	-589.0	-579.3	+/-	30.5	199	-195.0	-190.2	+/-	30.5	199	-191.0	-181.3	+/-	30.5
35	117	-239.0	-219.9	+/-	72.8	199	-129.0	-125.8	+/-	74.2	199	-35.0	-42.5	+/-	75.8
36	13	-35.0	-35.0	+/-	0.0	107	-81.0	-83.3	+/-	26.5	199	-115.0	-113.3	+/-	38.2
40	10	-30.0	-30.0	+/-	0.0	21	-21.0	-21.0	+/-	0.0	199	-199.0	-199.0	+/-	0.0
41	12	4.0	4.0	+/-	0.0	171	-117.0	-110.2	+/-	40.0	199	-103.0	-85.2	+/-	46.9
43	11	-25.0	-25.0	+/-	0.0	187	-127.0	-122.4	+/-	67.3	199	-67.0	-79.7	+/-	67.2
44	33	-75.0	-88.3	+/-	7.2	199	-163.0	-184.1	+/-	34.3	199	-103.0	-139.1	+/-	39.1
45	22	-58.0	-58.0	+/-	0.0	199	-157.0	-176.2	+/-	45.2	199	-91.0	-119.6	+/-	50.5
46	10	-22.0	-22.0	+/-	0.0	25	-19.0	-19.0	+/-	0.0	199	-167.0	-159.0	+/-	23.6
47	10	2.0	2.0	+/-	0.0	109	-67.0	-60.0	+/-	29.8	199	-91.0	-101.0	+/-	35.5
48	55	-149.0	-165.0	+/-	0.0	199	-189.0	-199.0	+/-	0.0	199	-183.0	-199.0	+/-	0.0
49	10	-14.0	-14.0	+/-	0.0	123	-53.0	-52.5	+/-	46.6	199	49.0	47.4	+/-	61.8
50	10	-30.0	-30.0	+/-	0.0	64	-64.0	-64.0	+/-	0.0	199	-195.0	-199.0	+/-	0.0
63	31	-37.0	-33.0	+/-	11.0	199	-141.0	-138.5	+/-	51.3	199	-95.0	-99.6	+/-	48.9
66	10	-2.0	-2.0	+/-	0.0	68	-20.0	-17.7	+/-	15.2	199	-35.0	-3.9	+/-	51.2
76	10	-22.0	-22.0	+/-	0.0	33	-27.0	-27.0	+/-	0.0	199	-123.0	-117.2	+/-	38.1
82	10	-26.0	-26.0	+/-	0.0	10	-8.0	-8.0	+/-	0.0	199	-147.0	-136.2	+/-	30.4
84	10	-18.0	-18.0	+/-	0.0	10	-4.0	-4.0	+/-	0.0	199	-171.0	-170.2	+/-	16.8
94	10	-14.0	-14.0	+/-	0.0	62	-22.0	-21.5	+/-	10.5	199	-91.0	-85.0	+/-	35.3
95	10	-22.0	-22.0	+/-	0.0	23	-17.0	-17.0	+/-	0.0	199	-115.0	-85.1	+/-	42.3
96	10	-18.0	-18.0	+/-	0.0	22	-10.0	-10.0	+/-	0.0	199	-115.0	-101.4	+/-	36.5
100	10	-2.0	-2.0	+/-	0.0	27	-1.0	-1.0	+/-	0.0	199	-27.0	-43.3	+/-	43.1
103	10	-26.0	-26.0	+/-	0.0	10	-8.0	-8.0	+/-	0.0	77	-73.0	-73.0	+/-	0.0
106	10	-26.0	-26.0	+/-	0.0	10	-8.0	-8.0	+/-	0.0	199	-183.0	-178.1	+/-	16.9
113	27	-69.0	-68.0	+/-	3.9	199	-173.0	-172.2	+/-	39.5	199	-155.0	-165.0	+/-	37.2
115	10	-30.0	-30.0	+/-	0.0	10	-10.0	-10.0	+/-	0.0	199	-195.0	-190.3	+/-	12.4
117	29	-39.0	-40.3	+/-	8.5	199	-87.0	-52.0	+/-	87.8	199	5.0	86.5	+/-	87.3
119	10	-26.0	-26.0	+/-	0.0	36	-34.0	-34.0	+/-	0.0	199	-191.0	-195.0	+/-	0.0
123	10	-14.0	-14.0	+/-	0.0	10	-2.0	-2.0	+/-	0.0	66	-10.0	-10.0	+/-	0.0
125	10	-22.0	-22.0	+/-	0.0	10	-8.0	-8.0	+/-	0.0	199	-187.0	-182.6	+/-	12.0
142	10	-30.0	-30.0	+/-	0.0	10	-10.0	-10.0	+/-	0.0	199	-179.0	-199.0	+/-	0.0
143	10	-18.0	-18.0	+/-	0.0	10	-4.0	-4.0	+/-	0.0	199	-163.0	-153.4	+/-	23.4
161	10	-30.0	-30.0	+/-	0.0	10	-10.0	-10.0	+/-	0.0	199	-199.0	-199.0	+/-	0.0
169	10	-30.0	-30.0	+/-	0.0	18	-18.0	-18.0	+/-	0.0	199	-191.0	-190.2	+/-	12.8
171	10	-30.0	-30.0	+/-	0.0	10	-10.0	-10.0	+/-	0.0	182	-182.0	-182.0	+/-	0.0
173	10	-18.0	-18.0	+/-	0.0	10	-4.0	-4.0	+/-	0.0	77	-49.0	-49.0	+/-	0.0
174	10	-30.0	-30.0	+/-	0.0	34	-34.0	-34.0	+/-	0.0	199	-183.0	-178.7	+/-	21.7
182	10	-30.0	-30.0	+/-	0.0	10	-10.0	-10.0	+/-	0.0	199	-163.0	-156.3	+/-	26.7
188	10	-30.0	-30.0	+/-	0.0	10	-10.0	-10.0	+/-	0.0	80	-80.0	-80.0	+/-	0.0
191	10	-26.0	-26.0	+/-	0.0	90	-64.0	-65.0	+/-	20.5	199	-83.0	-105.0	+/-	42.2

Summary Statistics			
Run Number	HNC–category A, manual		
Number of Topics			50
Total number of documents over all topics			
	Run 1	Run 2	Run 3
Retrieved:	250	1079	3930
Relevant:	6576	6576	6576
Rel_ret:	172	490	1249

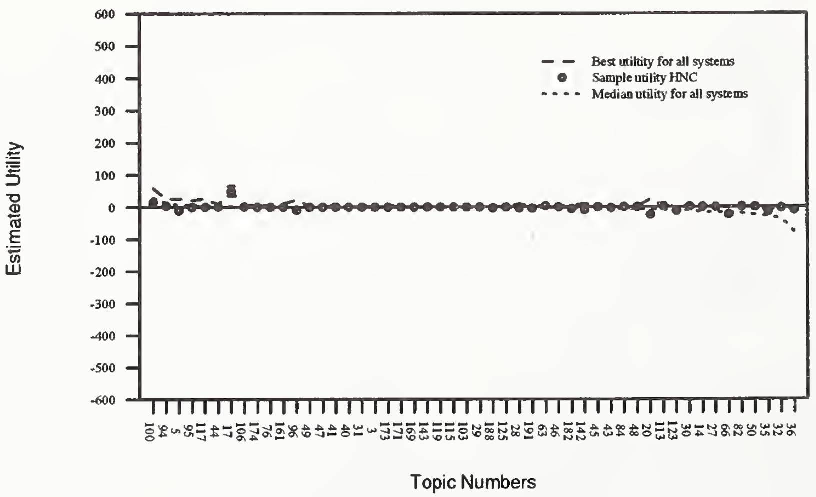
Average Effectiveness (Pooled Sample)

	Recall	Precision	Fallout
Run 1	0.020	0.259	0.0000
Run 2	0.059	0.326	0.0001
Run 3	0.171	0.342	0.0003

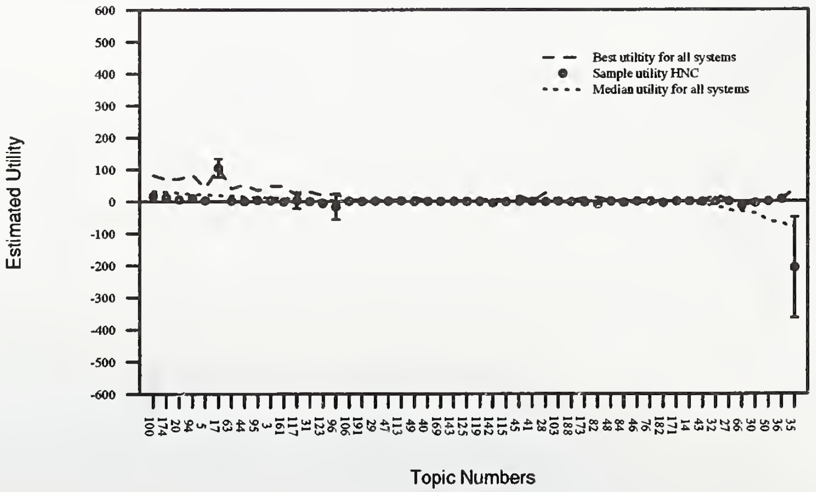
Number of topics where this site's Utility was

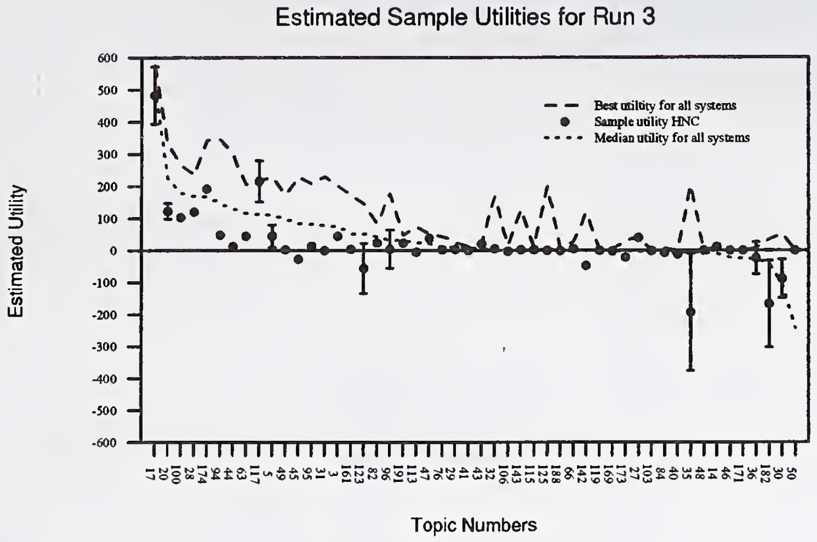
	1st	2nd	3rd	4th	Mean Rank
		Pooled			
Run 1	15	15	20	0	2.367
Run 2	13	12	21	4	2.143
Run 3	7	10	30	3	2.633
		Sampled			
Run 1	14	16	20	0	2.163
Run 2	13	14	19	4	2.327
Run 3	6	10	30	4	2.694

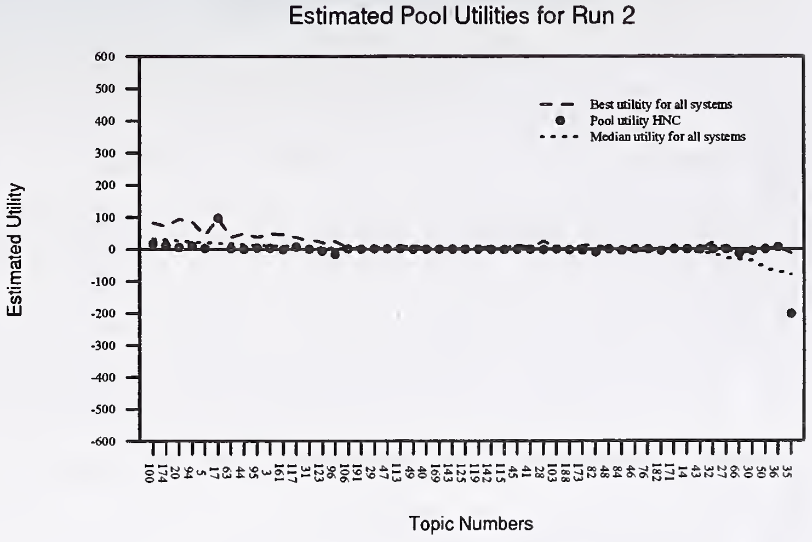
Estimated Sample Utilities for Run 1

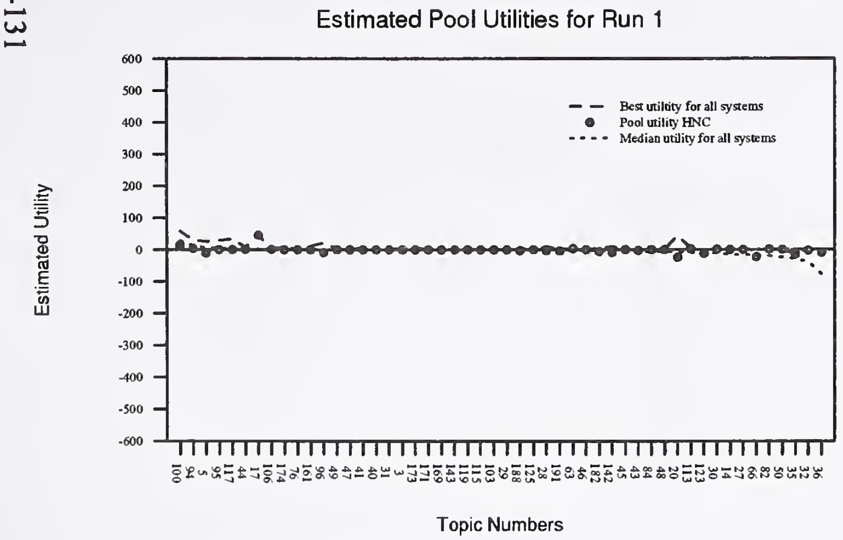


Estimated Sample Utilities for Run 2




Estimated Sample Utilities for Run 3
Estimated Utility
600
500
400
300
200
100
0
-100
-200
-300
-400
-500
-600
Best utility for all systems
Sample utility HNC
Median utility for all systems
17 20 100 28 174 94 44 63 117 5 49 45 95 31 3 161 123 82 96 191 113 47 76 29 41 43 32 106 143 115 125 188 66 142 119 169 173 27 103 84 40 35 48 14 46 171 36 182 30 50
Topic Numbers


Estimated Pool Utilities for Run 2
Estimated Utility
600
500
400
300
200
100
0
-100
-200
-300
-400
-500
-600
Best utility for all systems
Pool utility HNC
Median utility for all systems
100 174 20 94 5 17 63 44 95 3 161 117 31 123 96 106 191 29 47 113 49 40 169 143 125 119 142 115 45 41 28 103 188 173 82 48 84 46 76 182 171 14 43 32 27 66 30 50 36 35
Topic Numbers


Estimated Pool Utilities for Run 1
Estimated Utility
600
500
400
300
200
100
0
-100
-200
-300
-400
-500
-600
Best utility for all systems
Pool utility HNC
Median utility for all systems
100 94 5 95 117 44 17 106 174 76 161 96 49 47 41 40 31 3 173 171 169 143 119 115 103 29 188 125 28 191 63 46 182 142 45 43 84 48 20 113 123 30 14 27 66 82 50 35 32 36
Topic Numbers

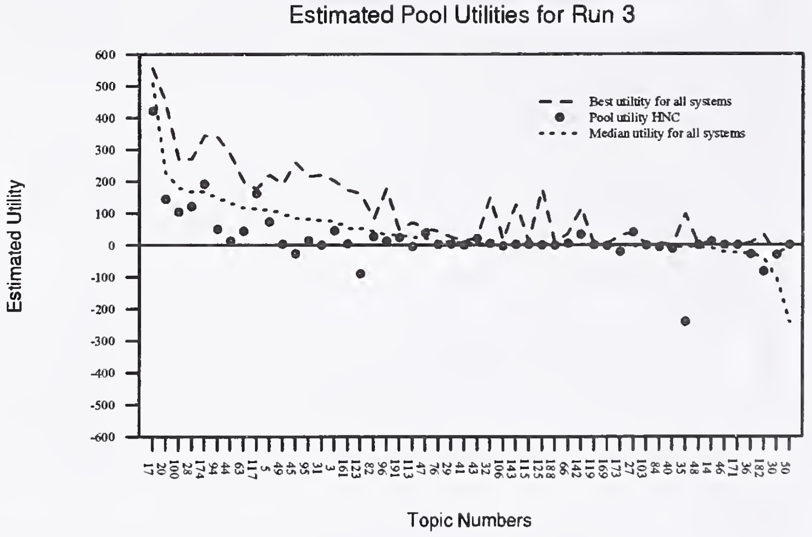

Estimated Pool Utilities for Run 3
Estimated Utility
600
500
400
300
200
100
0
-100
-200
-300
-400
-500
-600
Best utility for all systems
Pool utility HNC
Median utility for all systems
17 20 100 28 174 94 44 63 117 5 49 45 95 31 3 161 123 82 96 191 113 47 76 29 41 43 32 106 143 115 125 188 66 142 119 169 173 27 103 84 40 35 48 14 46 171 36 182 30 50
Topic Numbers

Table 1: Raw Data

topic	Run 1					Run 2					Run 3				
	set size	pool util.	sample util.			set size	pool util.	sample util.			set size	pool util.	sample util.		
3	0	0.0	0.0	+/-	0.0	25	3.0	3.0	+/-	0.0	55	45.0	45.0	+/-	0.0
5	17	-11.0	-11.0	+/-	0.0	17	3.0	3.0	+/-	0.0	158	74.0	45.1	+/-	35.5
14	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	9	11.0	11.0	+/-	0.0
17	66	46.0	50.0	+/-	15.4	133	97.0	104.7	+/-	28.3	274	422.0	483.4	+/-	88.5
20	36	-24.0	-24.0	+/-	0.0	62	6.0	6.0	+/-	0.0	124	144.0	122.9	+/-	24.7
27	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	40	40.0	40.0	+/-	0.0
28	1	-3.0	-3.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0	82	122.0	122.0	+/-	0.0
29	0	0.0	0.0	+/-	0.0	1	1.0	1.0	+/-	0.0	1	3.0	3.0	+/-	0.0
30	1	1.0	1.0	+/-	0.0	15	-5.0	-5.0	+/-	0.0	255	-31.0	-88.2	+/-	59.4
31	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
32	1	-3.0	-3.0	+/-	0.0	2	0.0	0.0	+/-	0.0	7	5.0	5.0	+/-	0.0
35	13	-15.0	-15.0	+/-	0.0	362	-202.0	-207.2	+/-	157.3	624	-240.0	-192.6	+/-	182.7
36	22	-10.0	-10.0	+/-	0.0	24	6.0	6.0	+/-	0.0	213	-29.0	-23.7	+/-	49.8
40	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	12	-12.0	-12.0	+/-	0.0
41	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
43	1	-3.0	-3.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0	48	20.0	20.0	+/-	0.0
44	1	1.0	1.0	+/-	0.0	6	0.0	0.0	+/-	0.0	15	13.0	13.0	+/-	0.0
45	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	35	-27.0	-27.0	+/-	0.0
46	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
47	0	0.0	0.0	+/-	0.0	7	1.0	1.0	+/-	0.0	62	38.0	38.0	+/-	0.0
48	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
49	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	1	3.0	3.0	+/-	0.0
50	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
63	3	3.0	3.0	+/-	0.0	18	2.0	2.0	+/-	0.0	43	45.0	45.0	+/-	0.0
66	13	-23.0	-23.0	+/-	0.0	42	-14.0	-14.0	+/-	0.0	71	5.0	5.0	+/-	0.0
76	0	0.0	0.0	+/-	0.0	2	0.0	0.0	+/-	0.0	10	2.0	2.0	+/-	0.0
82	5	1.0	1.0	+/-	0.0	51	-9.0	-9.0	+/-	0.0	102	26.0	23.6	+/-	4.6
84	0	0.0	0.0	+/-	0.0	4	-4.0	-4.0	+/-	0.0	47	-7.0	-7.0	+/-	0.0
94	4	4.0	4.0	+/-	0.0	16	10.0	10.0	+/-	0.0	26	50.0	50.0	+/-	0.0
95	0	0.0	0.0	+/-	0.0	4	4.0	4.0	+/-	0.0	6	14.0	14.0	+/-	0.0
96	27	-9.0	-9.0	+/-	0.0	116	-16.0	-16.1	+/-	40.7	251	13.0	4.2	+/-	59.9
100	14	14.0	14.0	+/-	0.0	17	17.0	17.0	+/-	0.0	40	104.0	104.0	+/-	0.0
103	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0
106	1	1.0	1.0	+/-	0.0	1	1.0	1.0	+/-	0.0	11	-3.0	-3.0	+/-	0.0
113	2	2.0	2.0	+/-	0.0	2	2.0	2.0	+/-	0.0	81	-5.0	-5.0	+/-	0.0
115	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0	2	2.0	2.0	+/-	0.0
117	0	0.0	0.0	+/-	0.0	77	7.0	3.1	+/-	25.5	210	162.0	216.2	+/-	64.1
119	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
123	12	-12.0	-12.0	+/-	0.0	28	-6.0	-6.0	+/-	0.0	286	-90.0	-55.7	+/-	78.4
125	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
142	3	-9.0	-9.0	+/-	0.0	5	-1.0	-5.0	+/-	0.0	47	33.0	-47.0	+/-	0.0
143	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	2	2.0	2.0	+/-	0.0
161	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0	4	4.0	4.0	+/-	0.0
169	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	2	-2.0	-2.0	+/-	0.0
171	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
173	0	0.0	0.0	+/-	0.0	5	-3.0	-3.0	+/-	0.0	41	-21.0	-21.0	+/-	0.0
174	0	0.0	0.0	+/-	0.0	14	12.0	12.0	+/-	0.0	72	192.0	192.0	+/-	0.0
182	2	-6.0	-6.0	+/-	0.0	5	-5.0	-5.0	+/-	0.0	502	-82.0	-167.2	+/-	135.3
188	1	-3.0	-3.0	+/-	0.0	2	-2.0	-2.0	+/-	0.0	9	-1.0	-1.0	+/-	0.0
191	4	-4.0	-4.0	+/-	0.0	12	0.0	0.0	+/-	0.0	49	23.0	23.0	+/-	0.0

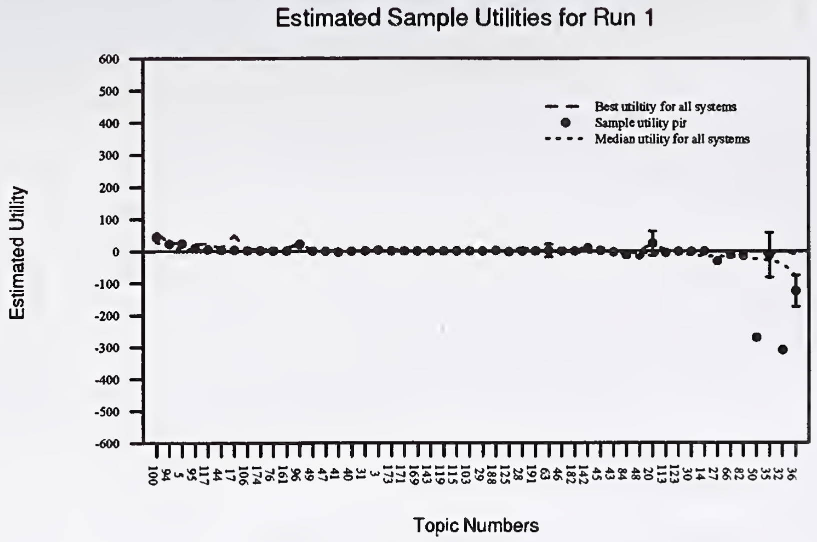
Summary Statistics			
Run Number	pir–category A, manual		
Number of Topics			50
Total number of documents over all topics			
	Run 1	Run 2	Run 3
Retrieved:	896	2707	6150
Relevant:	6576	6576	6576
Rel_ret:	556	1295	2625

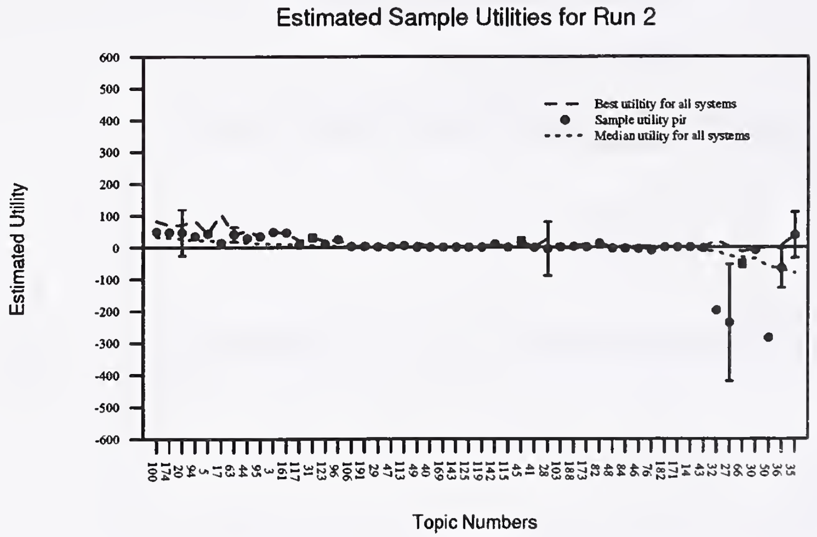
Average Effectiveness (Pooled Sample)

	Recall	Precision	Fallout
Run 1	0.085	0.464	0.0000
Run 2	0.182	0.474	0.0001
Run 3	0.341	0.455	0.0003

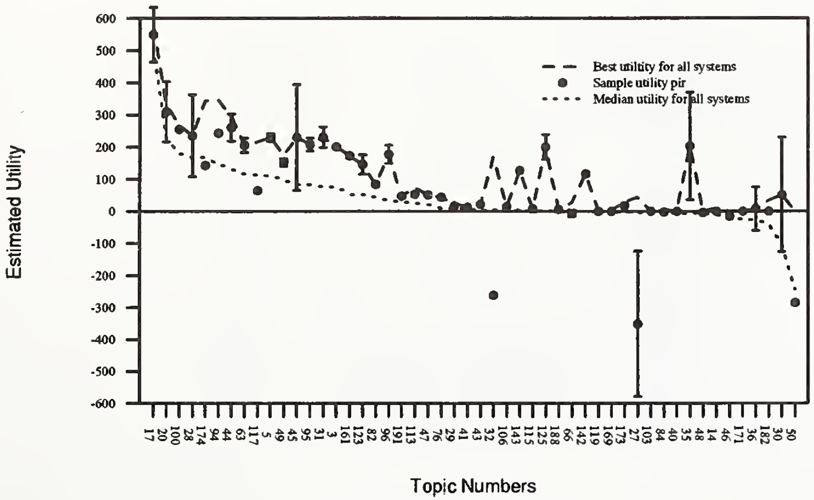
Number of topics where this site's Utility was

	1st	2nd	3rd	4th	Mean Rank
		Pooled			
Run 1	13	27	7	3	2.041
Run 2	20	21	6	3	1.878
Run 3	27	18	4	1	1.612
		Sampled			
Run 1	14	27	6	3	2.000
Run 2	20	20	6	4	1.918
Run 3	28	16	2	4	1.673

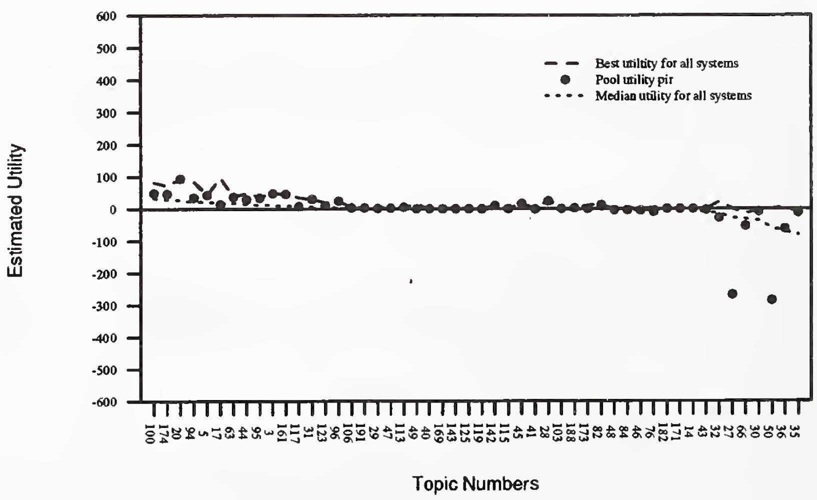




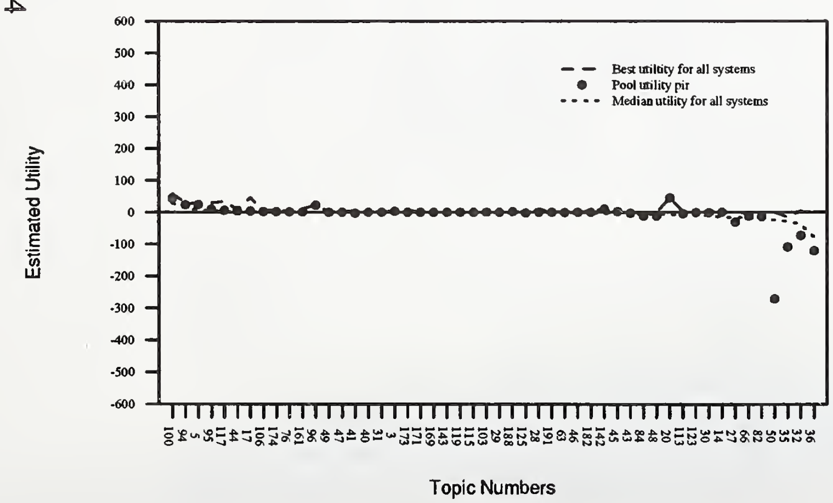
Estimated Sample Utilities for Run 3



Estimated Pool Utilities for Run 2



Estimated Pool Utilities for Run 1



Estimated Pool Utilities for Run 3

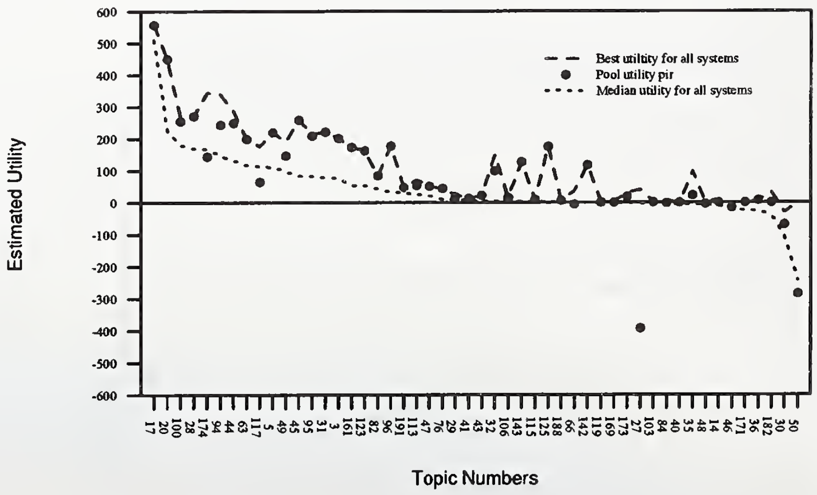


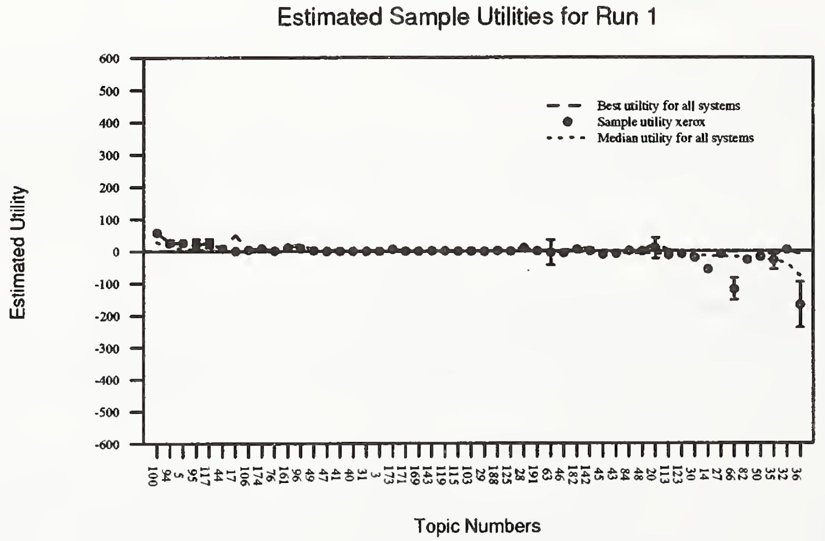
Table 1: Raw Data

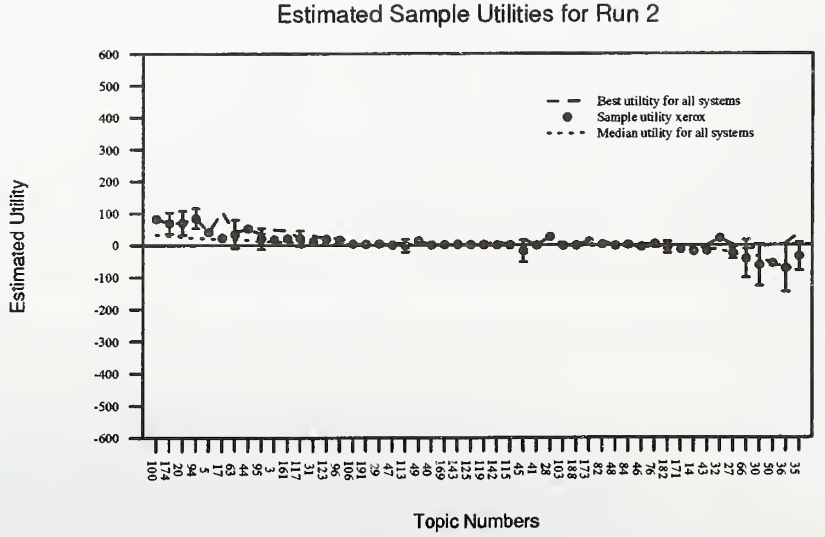
topic	Run 1					Run 2					Run 3				
	set size	pool util.	sample util.			set size	pool util.	sample util.			set size	pool util.	sample util.		
3	11	3.0	3.0	+/-	0.0	74	48.0	48.0	+/-	0.0	95	201.0	201.0	+/-	0.0
5	40	24.0	24.0	+/-	0.0	59	43.0	43.0	+/-	0.0	108	220.0	229.9	+/-	11.4
14	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
17	4	4.0	4.0	+/-	0.0	14	14.0	14.0	+/-	0.0	283	557.0	548.7	+/-	84.6
20	89	45.0	24.3	+/-	37.7	196	94.0	46.9	+/-	72.1	310	450.0	309.9	+/-	93.4
27	25	-31.0	-31.0	+/-	0.0	401	-267.0	-236.7	+/-	182.7	809	-393.0	-351.6	+/-	227.2
28	4	0.0	0.0	+/-	0.0	178	24.0	-5.3	+/-	84.6	381	271.0	235.2	+/-	127.7
29	0	0.0	0.0	+/-	0.0	15	1.0	1.0	+/-	0.0	15	17.0	17.0	+/-	0.0
30	3	-1.0	-1.0	+/-	0.0	47	-9.0	-9.0	+/-	0.0	445	-69.0	51.5	+/-	177.6
31	0	0.0	0.0	+/-	0.0	58	30.0	30.2	+/-	10.3	143	221.0	230.5	+/-	31.9
32	103	-73.0	-309.0	+/-	0.0	198	-28.0	-198.0	+/-	0.0	261	99.0	-261.0	+/-	0.0
35	131	-109.0	-11.9	+/-	69.9	180	-12.0	37.3	+/-	72.0	470	22.0	203.4	+/-	166.7
36	92	-120.0	-124.5	+/-	49.1	178	-62.0	-65.8	+/-	62.2	253	7.0	7.9	+/-	67.2
40	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
41	1	-3.0	-3.0	+/-	0.0	3	-1.0	-1.0	+/-	0.0	19	13.0	13.0	+/-	0.0
43	1	-3.0	-3.0	+/-	0.0	13	-3.0	-3.0	+/-	0.0	46	22.0	22.0	+/-	0.0
44	12	4.0	4.0	+/-	0.0	53	29.0	29.0	+/-	0.0	155	249.0	260.7	+/-	42.1
45	10	2.0	2.0	+/-	0.0	64	16.0	16.8	+/-	12.8	377	259.0	229.8	+/-	164.4
46	0	0.0	0.0	+/-	0.0	5	-5.0	-5.0	+/-	0.0	16	-16.0	-16.0	+/-	0.0
47	0	0.0	0.0	+/-	0.0	14	2.0	2.0	+/-	0.0	33	51.0	51.0	+/-	0.0
48	4	-12.0	-12.0	+/-	0.0	4	-4.0	-4.0	+/-	0.0	4	-4.0	-4.0	+/-	0.0
49	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	109	147.0	152.6	+/-	12.1
50	90	-270.0	-270.0	+/-	0.0	285	-285.0	-285.0	+/-	0.0	285	-285.0	-285.0	+/-	0.0
63	63	-1.0	1.5	+/-	19.8	111	37.0	40.5	+/-	22.8	129	199.0	205.9	+/-	22.8
66	12	-12.0	-12.0	+/-	0.0	101	-53.0	-52.9	+/-	10.6	110	-6.0	-5.9	+/-	10.6
76	1	1.0	1.0	+/-	0.0	56	-10.0	-10.0	+/-	0.0	68	44.0	44.0	+/-	0.0
82	38	-14.0	-14.0	+/-	0.0	38	12.0	12.0	+/-	0.0	64	84.0	84.0	+/-	0.0
84	4	-12.0	-12.0	+/-	0.0	6	-4.0	-4.0	+/-	0.0	6	-2.0	-2.0	+/-	0.0
94	23	23.0	23.0	+/-	0.0	41	35.0	35.0	+/-	0.0	97	243.0	243.0	+/-	0.0
95	14	10.0	10.0	+/-	0.0	42	34.0	34.0	+/-	0.0	119	209.0	208.2	+/-	19.7
96	22	22.0	22.0	+/-	0.0	24	24.0	24.0	+/-	0.0	135	177.0	177.2	+/-	28.2
100	48	44.0	44.0	+/-	0.0	51	49.0	49.0	+/-	0.0	89	255.0	255.0	+/-	0.0
103	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
106	2	2.0	2.0	+/-	0.0	5	3.0	3.0	+/-	0.0	13	15.0	15.0	+/-	0.0
113	15	-5.0	-5.0	+/-	0.0	15	5.0	5.0	+/-	0.0	39	53.0	53.0	+/-	0.0
115	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	12	8.0	8.0	+/-	0.0
117	10	6.0	6.0	+/-	0.0	10	8.0	8.0	+/-	0.0	44	64.0	64.0	+/-	0.0
119	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
123	7	-1.0	-1.0	+/-	0.0	32	10.0	10.0	+/-	0.0	138	162.0	145.5	+/-	30.3
125	2	-2.0	-2.0	+/-	0.0	2	0.0	0.0	+/-	0.0	161	175.0	199.9	+/-	38.9
142	10	10.0	10.0	+/-	0.0	10	10.0	10.0	+/-	0.0	39	117.0	117.0	+/-	0.0
143	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	73	127.0	127.0	+/-	0.0
161	1	1.0	1.0	+/-	0.0	58	46.0	46.0	+/-	0.0	91	173.0	173.0	+/-	0.0
169	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
171	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
173	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	10	18.0	18.0	+/-	0.0
174	2	2.0	2.0	+/-	0.0	49	47.0	47.0	+/-	0.0	49	143.0	143.0	+/-	0.0
182	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
188	2	2.0	2.0	+/-	0.0	2	2.0	2.0	+/-	0.0	2	6.0	6.0	+/-	0.0
191	0	0.0	0.0	+/-	0.0	15	3.0	3.0	+/-	0.0	45	47.0	47.0	+/-	0.0

Summary Statistics			
Run Number	xerox–category A, manual		
Number of Topics	50		
Total number of documents over all topics			
	Run 1	Run 2	Run 3
Retrieved:	941	2463	5735
Relevant:	6576	6576	6576
Rel_ret:	639	1405	2363

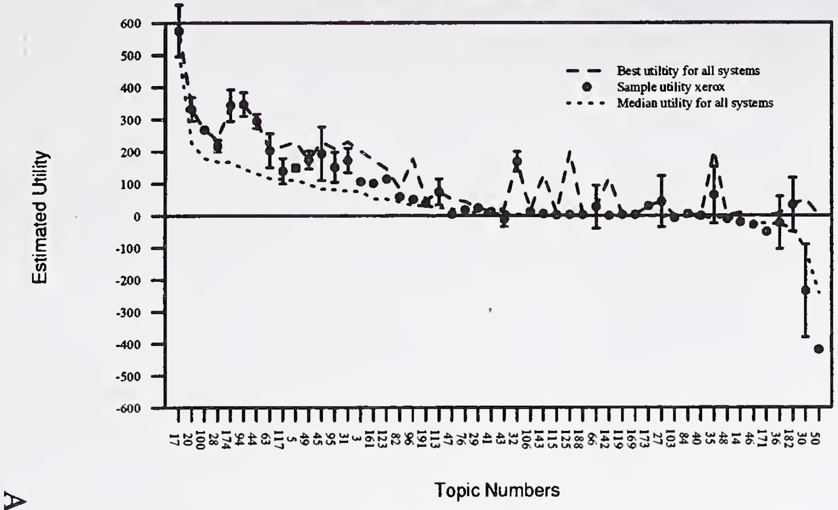
Average Effectiveness (Pooled Sample)			
	Recall	Precision	Fallout
Run 1	0.095	0.527	0.0000
Run 2	0.190	0.577	0.0001
Run 3	0.312	0.487	0.0003

Number of topics where this site's Utility was					
	1st	2nd	3rd	4th	Mean Rank
Pooled					
Run 1	21	16	9	4	1.959
Run 2	23	17	8	2	1.816
Run 3	18	21	9	2	1.939
Sampled					
Run 1	21	15	10	4	1.980
Run 2	21	18	9	2	1.878
Run 3	18	23	7	2	1.898

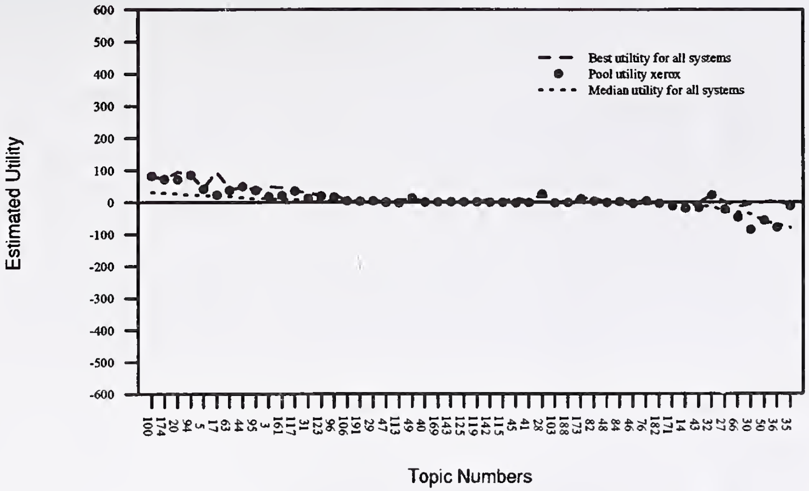

Estimated Sample Utilities for Run 1


Estimated Sample Utilities for Run 2

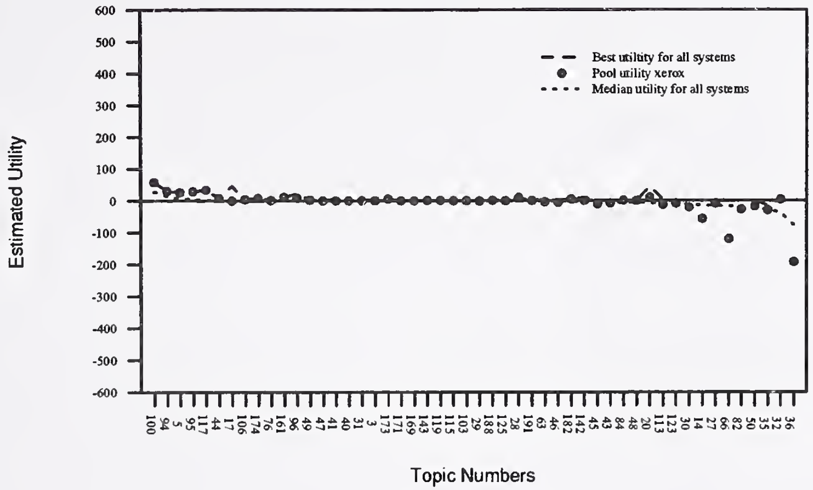
Estimated Sample Utilities for Run 3



Estimated Pool Utilities for Run 2



Estimated Pool Utilities for Run 1



Estimated Pool Utilities for Run 3

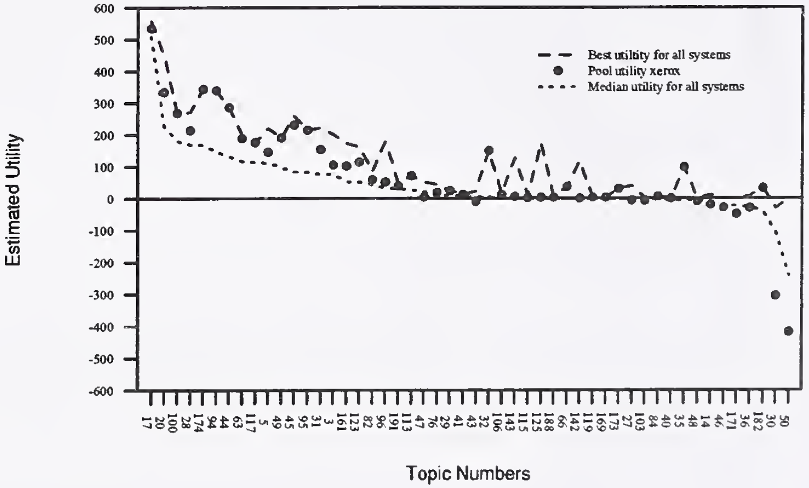


Table 1: Raw Data

	Run 1					Run 2					Run 3				
topic	set	pool	sample			set	pool	sample			set	pool	sample		
	size	util.	util.			size	util.	util.			size	util.	util.		
3	4	0.0	0.0	+/-	0.0	24	18.0	18.0	+/-	0.0	47	105.0	105.0	+/-	0.0
5	26	26.0	26.0	+/-	0.0	59	41.0	41.0	+/-	0.0	106	146.0	149.0	+/-	9.5
14	19	-57.0	-57.0	+/-	0.0	19	-19.0	-19.0	+/-	0.0	28	-20.0	-20.0	+/-	0.0
17	0	0.0	0.0	+/-	0.0	33	23.0	23.0	+/-	0.0	269	535.0	575.7	+/-	80.3
20	79	11.0	8.8	+/-	32.1	133	71.0	70.1	+/-	37.2	162	334.0	332.2	+/-	37.2
27	3	-9.0	-9.0	+/-	0.0	68	-22.0	-24.7	+/-	17.9	246	-6.0	44.1	+/-	79.5
28	14	10.0	10.0	+/-	0.0	44	26.0	26.0	+/-	0.0	118	214.0	217.6	+/-	18.3
29	3	-1.0	-1.0	+/-	0.0	6	4.0	4.0	+/-	0.0	28	24.0	24.0	+/-	0.0
30	11	-21.0	-21.0	+/-	0.0	154	-86.0	-63.5	+/-	64.9	561	-305.0	-235.3	+/-	144.5
31	0	0.0	0.0	+/-	0.0	20	12.0	12.0	+/-	0.0	155	153.0	172.2	+/-	38.0
32	8	4.0	4.0	+/-	0.0	44	22.0	22.0	+/-	0.0	138	150.0	168.6	+/-	31.5
35	62	-30.0	-28.2	+/-	28.3	144	-12.0	-35.3	+/-	44.1	294	98.0	64.6	+/-	88.2
36	124	-192.0	-167.8	+/-	71.0	191	-79.0	-72.6	+/-	74.2	298	-30.0	-22.4	+/-	82.2
40	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
41	0	0.0	0.0	+/-	0.0	7	-1.0	-1.0	+/-	0.0	36	12.0	12.0	+/-	0.0
43	4	-8.0	-8.0	+/-	0.0	43	-17.0	-17.0	+/-	0.0	134	-10.0	-11.8	+/-	23.0
44	12	8.0	8.0	+/-	0.0	63	49.0	51.7	+/-	6.5	130	286.0	294.4	+/-	21.9
45	17	-11.0	-11.0	+/-	0.0	98	-2.0	-18.7	+/-	34.0	245	231.0	193.5	+/-	83.2
46	2	-6.0	-6.0	+/-	0.0	12	-4.0	-4.0	+/-	0.0	69	-29.0	-29.0	+/-	0.0
47	0	0.0	0.0	+/-	0.0	2	0.0	0.0	+/-	0.0	4	4.0	4.0	+/-	0.0
48	0	0.0	0.0	+/-	0.0	1	-1.0	-1.0	+/-	0.0	14	-10.0	-10.0	+/-	0.0
49	2	2.0	2.0	+/-	0.0	37	13.0	13.0	+/-	0.0	133	191.0	174.2	+/-	27.9
50	6	-18.0	-18.0	+/-	0.0	57	-57.0	-57.0	+/-	0.0	419	-419.0	-419.0	+/-	0.0
63	84	-4.0	-4.9	+/-	39.0	135	37.0	34.9	+/-	44.3	203	189.0	202.7	+/-	52.9
66	71	-121.0	-118.3	+/-	33.9	161	-47.0	-42.8	+/-	59.2	252	36.0	27.3	+/-	67.1
76	1	1.0	1.0	+/-	0.0	8	4.0	4.0	+/-	0.0	10	18.0	18.0	+/-	0.0
82	45	-27.0	-27.0	+/-	0.0	57	3.0	3.0	+/-	0.0	61	59.0	59.0	+/-	0.0
84	1	1.0	1.0	+/-	0.0	4	2.0	2.0	+/-	0.0	30	6.0	6.0	+/-	0.0
94	42	30.0	26.7	+/-	7.7	119	85.0	84.2	+/-	30.9	169	339.0	346.5	+/-	36.9
95	53	29.0	20.9	+/-	16.2	115	37.0	20.0	+/-	32.6	193	215.0	150.5	+/-	47.0
96	9	9.0	9.0	+/-	0.0	16	16.0	16.0	+/-	0.0	17	51.0	51.0	+/-	0.0
100	58	58.0	58.0	+/-	0.0	84	82.0	82.0	+/-	0.0	100	268.0	268.0	+/-	0.0
103	0	0.0	0.0	+/-	0.0	2	-2.0	-2.0	+/-	0.0	7	-7.0	-7.0	+/-	0.0
106	4	4.0	4.0	+/-	0.0	4	4.0	4.0	+/-	0.0	4	12.0	12.0	+/-	0.0
113	27	-13.0	-13.0	+/-	0.0	80	-2.0	-1.8	+/-	19.6	165	71.0	74.0	+/-	40.8
115	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	2	2.0	2.0	+/-	0.0
117	50	34.0	25.8	+/-	13.2	103	35.0	20.2	+/-	25.0	176	176.0	139.1	+/-	39.8
119	1	1.0	1.0	+/-	0.0	1	1.0	1.0	+/-	0.0	1	3.0	3.0	+/-	0.0
123	39	-9.0	-9.0	+/-	0.0	65	19.0	19.0	+/-	0.0	86	114.0	114.0	+/-	0.0
125	0	0.0	0.0	+/-	0.0	1	1.0	1.0	+/-	0.0	1	3.0	3.0	+/-	0.0
142	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0	0	0.0	0.0	+/-	0.0
143	1	1.0	1.0	+/-	0.0	2	2.0	2.0	+/-	0.0	2	6.0	6.0	+/-	0.0
161	11	11.0	11.0	+/-	0.0	25	21.0	21.0	+/-	0.0	51	101.0	101.0	+/-	0.0
169	0	0.0	0.0	+/-	0.0	1	1.0	1.0	+/-	0.0	5	3.0	3.0	+/-	0.0
171	0	0.0	0.0	+/-	0.0	12	-12.0	-12.0	+/-	0.0	50	-50.0	-50.0	+/-	0.0
173	10	6.0	6.0	+/-	0.0	13	11.0	11.0	+/-	0.0	18	30.0	30.0	+/-	0.0
174	28	8.0	8.0	+/-	0.0	112	72.0	69.3	+/-	33.3	188	344.0	343.6	+/-	49.3
182	9	5.0	5.0	+/-	0.0	70	-4.0	-5.2	+/-	18.1	264	32.0	34.3	+/-	83.7
188	1	1.0	1.0	+/-	0.0	3	-1.0	-1.0	+/-	0.0	5	3.0	3.0	+/-	0.0
191	0	0.0	0.0	+/-	0.0	11	3.0	3.0	+/-	0.0	41	39.0	39.0	+/-	0.0

APPENDIX B

This appendix contains the system description forms filled out by each participating group. These forms are meant to supplement the system papers and contain a standard and formatted description of system features and timing aspects.

System Summary and Timing
Organization Name: Cornell University
List of Run ID's: CrnlAE CrnlAL CrnlRE CrnlRL CrnlI1 CrnlI2 CrnlBc10 CrnlB CrnlSE CrnlSV

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list: 592
- Controlled Vocabulary?: No
- Stemming Algorithm: SMART Lovin's based
- Term Weighting: Yes, tf * idf with new doc normalization
- Phrase Discovery?: yes
  - Kind of Phrase: Two adjacent non-stopwords occurring 25 times in D1
  - Method Used (statistical, syntactic, other): statistical
- Proper Noun Identification Algorithm?: yes
- Tokenizer?: yes, but results not used.

Statistics on Data Structures built from TREC Text

- Inverted index
  - Run ID: CrnlAE CrnlAL CrnlI1 CrnlI2
  - Total Storage (in MB): 731
  - Total Computer Time to Build (in hours): under 3.5
  - Automatic Process? (If not, number of manual hours): yes
  - Use of Term Positions?: no
  - Only Single Terms Used?: no
- Inverted index
  - Run ID: CrnlRE CrnlRL
  - Total Storage (in MB): ??
  - Total Computer Time to Build (in hours): 3
- Inverted index
  - Run ID: CrnlSE
  - Total Storage (in MB): 85
  - Total Computer Time to Build (in hours): under .5
  - Automatic Process? (If not, number of manual hours): yes
  - Use of Term Positions?: no
  - Only Single Terms Used?: yes
- Inverted index
  - Run ID: CrnlBc10
  - Total Storage (in MB): 610
  - Total Computer Time to Build (in hours): 1.2 (14 hours elapsed)
  - Automatic Process? (If not, number of manual hours): yes
  - Use of Term Positions?: no
  - Only Single Terms Used?: yes
- Clusters
- N-grams, Suffix arrays, Signature Files
- Knowledge Bases
  - Use of Manual Labor
- Special Routing Structures
- Other Data Structures built from TREC text
  - Run ID: CrnlBc10
  - Type of Structure: trie representation of corruption dictionary
  - Total Storage (in MB): 400
  - Total Computer Time to Build (in hours): 4 elapsed hours(but lots of

paging)

- - Automatic Process? (If not, number of manual hours): yes
  - Brief Description of Method: sort words in dictionary and formtrie.
- Other Data Structures built from TREC text
  - Run ID: CrnlAE CrnlAL CrnlI1 CrnlI2 CrnlRE CrnlRL CrnlBc10
  - Type of Structure: list of top occurring phrases in Disk 1
  - Total Storage (in MB): 1
  - Total Computer Time to Build (in hours): 4 elapsed hours
  - Automatic Process? (If not, number of manual hours): yes
  - Brief Description of Method: index and sort phrases by freq, keeping those occurring in 25 docs.

Data Built from Sources Other than the Input Text

- Internally-built Auxiliary File
  - Use of Manual Labor
- Externally-built Auxiliary File

Query construction

Automatically Built Queries (Ad-Hoc)

- Topic Fields Used: all
- Average Computer Time to Build Query (in cpu seconds): .02 seconds/query for Pass 1 query. Running a retrieval, reindexing top 20 docs, and expanding query using relevance feedback took .95 CPU seconds/query
- Method used in Query Construction
  - Term Weighting (weights based on terms in topics)?: yes
  - Phrase Extraction from Topics?: yes
  - Proper Noun Identification Algorithm?: not used
  - Tokenizer?:
    - Patterns which are Tokenized: not used
  - Expansion of Queries using Previously-Constructed Data Structure?:

Automatically Built Queries (Routing)

- Topic Fields Used: all
- Average Computer Time to Build Query (in cpu seconds): CrnlRE: .02 seconds to index original query. 48 seconds/query to gather relevance statistics (amortized and neglible in operational environment) and 240 seconds/query to form final query (almost all DFO). CrnlRL: .02 seconds to index original query. 48 seconds/query to gather statistics (amortized and neglible in operational environment) and 116 seconds/query to form final query.
- Method used in Query Construction
  - Terms Selected From
    - Topics: yes
    - All Training Documents: yes
  - Term Weighting with Weights Based on terms in
    - Topics: yes
    - All Training Documents: yes
    - Documents with Relevance Judgments: yes
  - Phrase Extraction from
    - Topics: yes (normal adjacency phrases for CrnlRE CrnlRL)
    - All Training Documents: yes
    - Documents with Relevance Judgments: CrnlRL added info about pairs of terms occurring closely together.
  - Syntactic Parsing

  - Word Sense Disambiguation using
  - Proper Noun Identification Algorithm from
  - Tokenizer
  - Heuristic Associations to Add Terms from
  - Expansion of Queries using Previously-Constructed Data Structure:
  - Automatic Addition of Boolean connectors or Proximity Operators using information from

Interactive Queries

- Initial Query Built Automatically or Manually: Automatically
- Type of Person doing Interaction
  - Domain Expert: no
  - System Expert: yes
- Average Time to do Complete Interaction
  - CPU Time (Total CPU Seconds for all Iterations): 14 seconds/query
  - Clock Time from Initial Construction of Query to Completion of Final Query (in minutes): CrnlI1:18, CrnlI2:14
- Average Number of Iterations: CrnlI1:4 CrnlI2 6.3
- Average Number of Documents Examined per Iteration: 10
- Minimum Number of Iterations: 2
- Maximum Number of Iterations: 9
- What Determines the End of an Iteration: Looked at all 10 docs presented
- Methods used in Interaction
  - Automatic Term Reweighting from Relevant Documents? : yes
  - Automatic Query Expansion from Relevant Documents? : yes
    - Only Top X Terms Added (what is X): 60
  - Other Automatic Methods: None. Only user input allowed is relevance judgement
  - Manual Methods

Searching

Search Times

  - Run ID: CrnlAE
  - Computer Time to Search (Average per Query, in CPU seconds): <28 seconds/query
  - Component Times: 1 second/query search, 27 seconds keeping track of top 1000
- Search Times
  - Run ID: CrnlAL
- Search Times
  - Run ID: CrnlSV
  - Computer Time to Search (Average per Query, in CPU seconds): .05 search plus 22 to keep track of top 1000 docs

Machine Searching Methods

  - Vector Space Model?: yes
  - Probabilistic Model?: some components

Factors in Ranking

  - Term Frequency?: yes
  - Inverse Document Frequency?: yes
  - Other Term Weights?: yes, eg CrnlAE do an initial retrieval and weight terms according to how often they occur in top 20 docs. The ITL runs (adhoc +

routing) used distance between terms in both query and doc.

  - Proximity of Terms?: yes
  - Document Length?: yes

Machine Information

- Machine Type for TREC Experiment: Sun SPARC 20/512
- Was the Machine Dedicated or Shared: dedicated
- Amount of Hard Disk Storage (in MB): 27,000
- Amount of RAM (in MB): 192

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: several years, mostly re-engineering
- Given appropriate resources
  - Could your system run faster?: yes
  - By how much (estimate)?: ??

System Summary and Timing
Organization Name: U. of California, Berkeley
List of Run ID's: Brkly9, Brkly10, Brkly11, Brkly12

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list: 592
- Controlled Vocabulary? : NO
- Stemming Algorithm: SMART STEMMER
  - Morphological Analysis: NO
- Term Weighting: YES
- Phrase Discovery? : NO
- Syntactic Parsing? : NO
- Word Sense Disambiguation? : NO
- Heuristic Associations (including short definition)? : NO
- Spelling Checking (with manual correction)? : NO
- Spelling Correction? : NO
- Proper Noun Identification Algorithm? : NO
- Tokenizer? : NO
- Manually-Indexed Terms? : NO
- Other Techniques for building Data Structures: NONE

Statistics on Data Structures built from TREC Text

- Inverted index
  - Run ID : Brkly9, Brkly10
  - Total Storage (in MB): 550
  - Total Computer Time to Build (in hours): APPROX. 50
  - Automatic Process? (If not, number of manual hours): YES
  - Use of Term Positions? : NO
  - Only Single Terms Used? : YES
- Inverted index
  - Run ID : Brkly11, Brkly12
  - Total Storage (in MB): 250
  - Total Computer Time to Build (in hours): APPROX. 25
  - Automatic Process? (If not, number of manual hours): YES
  - Use of Term Positions? : NO
  - Only Single Terms Used? : YES
- Clusters
- N-grams, Suffix arrays, Signature Files
- Knowledge Bases
  - Use of Manual Labor
- Special Routing Structures
- Other Data Structures built from TREC text

Query construction

Automatically Built Queries (Ad-Hoc)

- Topic Fields Used: DESCRIPTION
- Average Computer Time to Build Query (in cpu seconds): APPROX. 3
- Method used in Query Construction
  - Term Weighting (weights based on terms in topics)? : YES
  - Tokenizer? :
  - Expansion of Queries using Previously-Constructed Data Structure? :

Automatically Built Queries (Routing)

- Topic Fields Used: DOM, TITLE, DESC, NARR, CON, DEF, NAT TIME
- Average Computer Time to Build Query (in cpu seconds): APPROX. 50
- Method used in Query Construction
  - Terms Selected From
    - Topics: YES
    - Only Documents with Relevance Judgments: YES
  - Term Weighting with Weights Based on terms in
  - Phrase Extraction from
  - Syntactic Parsing
  - Word Sense Disambiguation using
  - Proper Noun Identification Algorithm from
  - Tokenizer
  - Heuristic Associations to Add Terms from
  - Expansion of Queries using Previously-Constructed Data Structure:
  - Automatic Addition of Boolean connectors or Proximity Operators using information from

Manually Constructed Queries (Ad-Hoc)

- Topic Fields Used: DESCRIPTION
- Average Time to Build Query (in Minutes): 25
- Type of Query Builder
- Tools used to Build Query
  - Knowledge Base Browser? :
  - Other Lexical Tools? :
- Method used in Query Construction
  - Addition of Terms not Included in Topic? :
    - Source of Terms: Boolean lookups in parallel collections

Searching

Search Times

- Run ID : Brkly9, Brkly10
- Computer Time to Search (Average per Query, in CPU seconds): APPROX. 30

Search Times

- Run ID : Brkly11, Brkly12
- Computer Time to Search (Average per Query, in CPU seconds): APPROX. 50

Machine Searching Methods

- Probabilistic Model? : YES

Factors in Ranking

- Term Frequency? : YES
- Inverse Document Frequency? : YES

Machine Information

- Machine Type for TREC Experiment: SPARC 10, SPARC 20
- Was the Machine Dedicated or Shared: SHARED
- Amount of Hard Disk Storage (in MB): 5,000
- Amount of RAM (in MB): 128
- Clock Rate of CPU (in MHz): 50 for SPARC 10, 90 for SPARC 20

System Summary and Timing
Organization Name: City University
List of Run ID's: citya1, citym1, cityr1, cityr2, cityi1

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list:
  - citya1 citym1 cityi1
  - cityr1 cityr2
- Controlled Vocabulary? : No
- Stemming Algorithm: Modified Porter with spelling normalization
  - Morphological Analysis: No
- Term Weighting: No
- Phrase Discovery? : No
- Syntactic Parsing? : No
- Word Sense Disambiguation? : No
- Heuristic Associations (including short definition)? : No
- Spelling Checking (with manual correction)? : No
- Spelling Correction? : No
- Proper Noun Identification Algorithm? : No
- Tokenizer? : No
- Manually-Indexed Terms? : No
- Other Techniques for building Data Structures: None

Statistics on Data Structures built from TREC Text

- Inverted index
  - Run ID : citya1 citym1 cityi1
  - Total Storage (in MB): 935
  - Total Computer Time to Build (in hours): ~~20
  - Automatic Process? (If not, number of manual hours): Yes
  - Use of Term Positions? : Yes
  - Only Single Terms Used? : Mainly
- Inverted index
  - Run ID : cityr1 cityr2
  - Total Storage (in MB): 395
  - Total Computer Time to Build (in hours): ~~8
  - Automatic Process? (If not, number of manual hours): Yes
  - Use of Term Positions? : Yes
  - Only Single Terms Used? : Mainly
- Clusters
- N-grams, Suffix arrays, Signature Files
- Knowledge Bases
  - Use of Manual Labor
- Special Routing Structures
- Other Data Structures built from TREC text

Data Built from Sources Other than the Input Text

- Internally-built Auxiliary File
  - Domain (independent or specific): Somewhat angled towards American data
  - Type of File (thesaurus, knowledge base, lexicon, etc.): Contains synonym classes go phrases stop- and semi-stop terms, prefixes
- Total Storage (in MB): <<1
  - Number of Concepts Represented: about 900
  - Type of Representation: Lookup table

  - Total Manual Time to Build (in hours): Not known, built piecemeal
  - Use of Manual Labor
    - Other: Yes
- Externally-built Auxiliary File

Query construction

Automatically Built Queries (Ad-Hoc)

- Topic Fields Used: DESCRIPTION
- Average Computer Time to Build Query (in cpu seconds): Several minutes
- Method used in Query Construction
  - Term Weighting (weights based on terms in topics)? : See below
  - Tokenizer? :
  - Expansion of Queries using Previously-Constructed Data Structure? :
  - Other: Queries built from terms extracted from documents retrieved by a pilot search using the topic statement. Weights based on statistics of occurrence in documents retrieved in pilot search, in topic statement, and in the collection. See text.

Automatically Built Queries (Routing)

- Topic Fields Used: TCND (but only to modify weights)
- Average Computer Time to Build Query (in cpu seconds): ~~ 2 hours
- Method used in Query Construction
  - Terms Selected From
    - Only Documents with Relevance Judgments: All documents with positive relevance judgments
  - Term Weighting with Weights Based on terms in
    - Topics: The weights of topic terms were modified if they occurred more than once in the topic statement.
    - Documents with Relevance Judgments: All docs with positive relevance judgments
  - Phrase Extraction from
  - Syntactic Parsing
  - Word Sense Disambiguation using
  - Proper Noun Identification Algorithm from
  - Tokenizer
  - Heuristic Associations to Add Terms from
  - Expansion of Queries using Previously-Constructed Data Structure:
  - Automatic Addition of Boolean connectors or Proximity Operators using information from

Manually Constructed Queries (Ad-Hoc)

- Topic Fields Used: DESCRIPTION
- Average Time to Build Query (in Minutes): Not recorded, a few minutes.
- Type of Query Builder
  - Domain Expert: No
  - Computer System Expert: Possibly
- Tools used to Build Query
  - Knowledge Base Browser? :
  - Other Lexical Tools? :
- Method used in Query Construction
  - Addition of Terms not Included in Topic? : Yes
    - Source of Terms: See below
  - Other: Procedure the same as the automatic ad hoc except that some query terms were manually removed both from the pilot query and also from the final query resulting from expansion.

Interactive Queries

- Initial Query Built Automatically or Manually: Manually
- Type of Person doing Interaction
- Average Time to do Complete Interaction
  - CPU Time (Total CPU Seconds for all Iterations): Unknown
  - Clock Time from Initial Construction of Query to Completion of Final Query (in minutes): 30
- Average Number of Iterations: 4.2
- Average Number of Documents Examined per Iteration: 7.5
- Minimum Number of Iterations: 1
- Maximum Number of Iterations: 8
- What Determines the End of an Iteration: a user initiated search on the current query formulation
- Methods used in Interaction
  - Automatic Term Reweighting from Relevant Documents? : Yes
  - Automatic Query Expansion from Relevant Documents? : Yes
    - All Terms in Relevant Documents added: Yes, but only the top 50 of the ranked list of reweighted terms available to the user
    - Only Top X Terms Added (what is X): 20 after possible removal of terms by the user
    - User Selected Terms Added: User could remove terms
  - Manual Methods
    - Using Individual Judgment (No Set Algorithm)? : Yes

Searching

Search Times

- Run ID : citya1 citym1 cityr1 cityr2
- Computer Time to Search (Average per Query, in CPU seconds): < 60
- Component Times : Usually about 20 seconds (mainly disk wait) to fetch the 1000 documents (we don't have a separate file of DOCNOs); the rest is initializing and searching
- Run ID : cityi1
- Computer Time to Search (Average per Query, in CPU seconds): Unknown
- Component Times : Unknown

Machine Searching Methods

- Probabilistic Model? : Yes

Factors in Ranking

- Term Frequency? : Yes
- Inverse Document Frequency? : Yes
- Other Term Weights? : From relevance or pseudo-relevance information when available.
- Document Length? : Yes

Machine Information

- Machine Type for TREC Experiment: SS10, IPX, 4/330, IPC; all running 4.1.3
- Was the Machine Dedicated or Shared: SS10 dedicated, others shared
- Amount of Hard Disk Storage (in MB): About 20GB
- Amount of RAM (in MB): 256, 48, 40, 8 (resp.)
- Clock Rate of CPU (in MHz): Various, don't know

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: A lot, over the years; more hacking together than software engineering.
- Given appropriate resources
  - Could your system run faster? : Yes
  - By how much (estimate)? : Up to two orders of magnitude, given the hardware. With current hardware, possibly factor of 2
- Features the System is Missing that would be beneficial: Might include
  (1) better parsing including phrase determination and/or recognition;
  (2) better model for processing sub-documents---there are a number of bodges in the way we do this at present, we need better methods for determining parameters (of which there are at least 8) and some way of using information on more than one sub-document of a given document.

Significant Areas of System

- Brief Description of features in your system which you feel impact the system and are not answered by above questions: All our runs made use of a facility for searching sub-documents.

System Summary and Timing
Organization Name: Xerox PARC
List of Run ID's: xerox1 xerox2

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Stemming Algorithm:
- Term Weighting: mixed: no weighting / LSI
- Phrase Discovery?:
  - Kind of Phrase: two-word phrases
  - Method Used (statistical, syntactic, other): statistical
- Syntactic Parsing?: no
- Word Sense Disambiguation?: no
- Heuristic Associations (including short definition)?: no
- Spelling Checking (with manual correction)?: no
- Spelling Correction?: no
- Proper Noun Identification Algorithm?: no
- Tokenizer?:
- Manually-Indexed Terms?: no
- Other Techniques for building Data Structures: none

Statistics on Data Structures built from TREC Text

- Inverted index
- Clusters
- N-grams, Suffix arrays, Signature Files
- Knowledge Bases
  - Use of Manual Labor
- Special Routing Structures
  - Run ID: xerox1 xerox2
  - Type of Structure: lsi
  - Total Storage (in MB): 40
  - Total Computer Time to Build (in hours): 5
  - Automatic Process? (If not, number of manual hours): yes
  - Brief Description of Method: local LSI, one for each topic on 2000 chisquare selected terms
- Other Data Structures built from TREC text

Data Built from Sources Other than the Input Text

- Internally-built Auxiliary File
  - Use of Manual Labor
- Externally-built Auxiliary File

Query construction

Automatically Built Queries (Ad-Hoc)

- Method used in Query Construction
  - Tokenizer?:

  - Expansion of Queries using Previously-Constructed Data Structure?:

Automatically Built Queries (Routing)

- Topic Fields Used: all fields

- Average Computer Time to Build Query (in cpu seconds): less than 5
- Method used in Query Construction
  - Terms Selected From
    - Topics: yes
    - All Training Documents: yes
    - Only Documents with Relevance Judgments: yes
  - Term Weighting with Weights Based on terms in
    - Topics: no weights or lsi weights
    - All Training Documents: no weights or lsi weights
    - Documents with Relevance Judgments: no weights or lsi weights
  - Phrase Extraction from
    - Topics: yes
    - All Training Documents: yes
    - Documents with Relevance Judgments: yes
  - Syntactic Parsing
    - Topics: no
    - All Training Documents: no
    - Documents with Relevance Judgments: no
  - Word Sense Disambiguation using
    - Topics: no
    - All Training Documents: no
    - Documents with Relevance Judgments: no
  - Proper Noun Identification Algorithm from
    - Topics: no
    - All Training Documents: no
    - Documents with Relevance Judgments: no
  - Tokenizer
  - Heuristic Associations to Add Terms from
    - Topics: no
    - All Training Documents: no
    - Documents with Relevance Judgments: no
  - Expansion of Queries using Previously-Constructed Data Structure:
  - Automatic Addition of Boolean connectors or Proximity Operators using information from

Searching

Machine Searching Methods

- Vector Space Model? : yes
- Probabilistic Model? : yes
- Neural Networks? : yes

Factors in Ranking

- Term Frequency? : yes
- Inverse Document Frequency? : yes
- Other Term Weights? : lsi
- Document Length? : yes

Machine Information

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: little
- Given appropriate resources
  - Could your system run faster?: yes
  - By how much (estimate)?: factor of 10

System Summary and Timing
Organization Name: Siemens Corporate Research, Inc.
List of Run ID's: siems1, siems2, siems3

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list: 571
- Stemming Algorithm: standard SMART
- Term Weighting: yes (cosine-normalized tf)
- Phrase Discovery? : yes
  - Kind of Phrase: two words
  - Method Used (statistical, syntactic, other): statistical
- Tokenizer? :

Statistics on Data Structures built from TREC Text

- Inverted index
  - Run ID : siems1
  - Total Storage (in MB): 727 MB
  - Total Computer Time to Build (in hours): 13.75
  - Automatic Process? (If not, number of manual hours): yes
  - Use of Term Positions? : no
  - Only Single Terms Used? : no
- Inverted index
  - Run ID : siems2, siems3
  - Total Storage (in MB): 10 individual indexes totaling 955 MB
  - Total Computer Time to Build (in hours): 20 hours total
  - Automatic Process? (If not, number of manual hours): yes
  - Use of Term Positions? : no
  - Only Single Terms Used? : no
- Clusters
- N-grams, Suffix arrays, Signature Files
- Knowledge Bases
  - Use of Manual Labor
- Special Routing Structures
- Other Data Structures built from TREC text

Data Built from Sources Other than the Input Text

- Internally-built Auxiliary File
  - Domain (independent or specific): collection specific
  - Type of File (thesaurus, knowledge base, lexicon, etc.):
    siems2: training query clusters
    siems3: ranks of relevant docs in training queries
  - Total Storage (in MB): siems2: ~ .8 per db siems3: ~ 1 per db
- Total Computer Time to Build (in hours): siems2: ~ 22 hours to do training query retrieval +~ .75 to cluster siems3: ~ 22 hours to do training query retrieval +~ .5 to make query collection
  - Use of Manual Labor
- Externally-built Auxiliary File
  - Type of File (Treebank, WordNet, etc.): list of phrases generated by Cornell from disk 1
  - Total Storage (in MB): 2
  - Number of Concepts Represented: 158099

Query construction

Automatically Built Queries (Ad-Hoc)

- Topic Fields Used: description
- Average Computer Time to Build Query (in cpu seconds): siems1: ~ .3 CPU secs siems2: .2 CPU secs ave. per query per database siems3: .5 CPU secs aver per query (in Query db)
- Method used in Query Construction
  - Term Weighting (weights based on terms in topics)? : yes
  - Phrase Extraction from Topics? : yes
  - Tokenizer? :
  - Expansion of Queries using Previously-Constructed Data Structure? :

Searching

Search Times

- Run ID : siems1
- Computer Time to Search (Average per Query, in CPU seconds): 74
- Component Times : search consists of: run initial query get relevance assessments construct expanded query run expanded query (did not time individual pieces, sorry)

- Search Times
  - Run ID : siems2
  - Computer Time to Search (Average per Query, in CPU seconds): 73 (assuming parallel searching of dbs)
  - Component Times : 72 CPU secs average per query per database searching plus .9 CPU secs average per query for actual merging
- Search Times
  - Run ID : siems3
  - Computer Time to Search (Average per Query, in CPU seconds): 103 (assuming parallel searching of dbs)
  - Component Times : 72 CPU secs average per query per database searching plus 31 CPU secs average per query for actual merging

Machine Searching Methods

- Vector Space Model? : yes
- Cluster Searching? : only in siems2 (for query clusters)

Factors in Ranking

- Term Frequency? : yes
- Inverse Document Frequency? : yes
- Document Length? : yes (cosine)
- Other: probabilistic creation of ranked set for merging runs siems2 and siems3. Document for next rank is randomly selected from dbs, with selection biased by the number of documents in each db remaining to be added to final ranking.

Machine Information

- Machine Type for TREC Experiment: SPARC-10/41
- Was the Machine Dedicated or Shared: mostly dedicated
- Amount of Hard Disk Storage (in MB): ~ 13,000
- Amount of RAM (in MB): 128
- Clock Rate of CPU (in MHz): 40 MHz

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: fusion code completely experimental. Retrieval done by SMART, a well-tuned research prototype.
- Given appropriate resources
  - Could your system run faster? : yes
  - By how much (estimate)? : at least halved for MRDD fusion: need to approximate optimization problem

System Summary and Timing
Organization Name: Queens College, City University of New York
List of Run ID's: Pircs1, Pircs2, PircsL, PircsC

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list: 630
- Stemming Algorithm: Porter's algorithm
- Term Weighting: yes
- Phrase Discovery? :
  - Kind of Phrase: 2-word
  - Method Used (statistical, syntactic, other): statistical
- Tokenizer? :
- Manually-Indexed Terms? : yes for pircs2
- Other Techniques for building Data Structures:011011

Statistics on Data Structures built from TREC Text

- Inverted index
- Clusters
- N-grams, Suffix arrays, Signature Files
- Knowledge Bases
  - Use of Manual Labor
- Special Routing Structures
  - Run ID : pircsL, pircsC
  - Type of Structure: Network of linked NODE and EDGE files capturing the query expansion terms and learnt weights built dynamically
  - Total Storage (in MB): about 90MB for 10 queries per 500MB textbase
  - Total Computer Time to Build (in hours): 2
  - Automatic Process? (If not, number of manual hours): yes
  - Brief Description of Method: built from direct files of queries and documents and known relevant document information
- Other Data Structures built from TREC text
  - Run ID : Pircs1,pircs2,pircsL,pircsC
  - Type of Structure: Compressed, truncated direct file; network of linked NODE and EDGE files built during query time
  - Total Storage (in MB): direct file - about 80MB per 500MB raw text; network - about 60MB for 10 queries per 500MB textbase
  - Total Computer Time to Build (in hours): direct file - about 20 minutes; network - about 5 min per 10 query
  - Automatic Process? (If not, number of manual hours): Yes
  - Brief Description of Method: built from direct files of queries and documents

Data Built from Sources Other than the Input Text

- Internally-built Auxiliary File
  - Domain (independent or specific): independent
  - Type of File (thesaurus, knowledge base, lexicon, etc.): Stop words
  - Total Storage (in MB): .004
  - Number of Concepts Represented: 630
  - Type of Representation: array
  - Total Computer Time to Modify for TREC (if already built): none already exists
  - Use of Manual Labor
- Internally-built Auxiliary File

    - Domain (independent or specific): independent
    - Type of File (thesaurus, knowledge base, lexicon, etc.): 2-word Phrases
    - Total Storage (in MB): .005
    - Number of Concepts Represented: 55599
    - Type of Representation: array
    - Total Computer Time to Build (in hours): already exists
    - Total Computer Time to Modify for TREC (if already built):
    - Use of Manual Labor
  - Externally-built Auxiliary File

Query construction

Automatically Built Queries (Ad-Hoc)

- Topic Fields Used: desc
- Average Computer Time to Build Query (in cpu seconds): 3 sec
- Method used in Query Construction
  - Term Weighting (weights based on terms in topics)? : yes
  - Phrase Extraction from Topics? :yes, 2-word
  - Tokenizer? :
  - Expansion of Queries using Previously-Constructed Data Structure? : yes
    - Structure Used: Network
  - Automatic Addition of Boolean Connectors or Proximity Operators? :

Automatically Built Queries (Routing)

- Topic Fields Used: title, desc, narr, con
- Average Computer Time to Build Query (in cpu seconds): 18
- Method used in Query Construction
  - Terms Selected From
    - Topics: yes
    - Only Documents with Relevance Judgments: yes
  - Term Weighting with Weights Based on terms in
    - Documents with Relevance Judgments: yes
  - Phrase Extraction from
    - Topics: yes
    - Documents with Relevance Judgments: yes
  - Syntactic Parsing
  - Word Sense Disambiguation using
  - Proper Noun Identification Algorithm from
  - Tokenizer
  - Heuristic Associations to Add Terms from
  - Expansion of Queries using Previously-Constructed Data Structure:
    - Structure Used: Network
  - Automatic Addition of Boolean connectors or Proximity Operators using information from

Manually Constructed Queries (Ad-Hoc)

- Topic Fields Used: desc
- Average Time to Build Query (in Minutes): about 3 hrs for 50 queries
- Type of Query Builder
  - Computer System Expert: yes
- Tools used to Build Query
  - Knowledge Base Browser? :
  - Other Lexical Tools? :
- Method used in Query Construction
  - Term Weighting? : yes

- Addition of Terms not Included in Topic? :
  - Source of Terms: human knowledge

Searching

Search Times

- Run ID :PircsL,PircsC
- Computer Time to Search (Average per Query, in CPU seconds): about 4 min clock time for pircsL, 3 times this for pircsC.
- Component Times : Build network 4 min (per 10 query) Retrieval 33 min (per 10 query) Sort, merge reformat results 3 min

Machine Searching Methods

- Probabilistic Model? : yes
- Boolean Matching? : yes
- Neural Networks? : yes

Factors in Ranking

- Term Frequency? : yes
- Other Term Weights? : yes, within-doc term frequency, inverse collection term frequency.
- Proximity of Terms? : yes, 2 word phrases
- Document Length? : yes

Machine Information

- Machine Type for TREC Experiment:Sparc 10 model 30
- Was the Machine Dedicated or Shared: Dedicated
- Amount of Hard Disk Storage (in MB): 7000
- Amount of RAM (in MB): 128

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: some space and time efficiency factors were made
- Given appropriate resources
  - Could your system run faster? :yes
  - By how much (estimate)? : probably half the time.
- Features the System is Missing that would be beneficial: ability to differentiate contexts

Significant Areas of System

- Brief Description of features in your system which you feel impact the system and are not answered by above questions:
  1. handles varied length documents by segmenting into subdocuments of about 550 words;
  2. training with subdocuments in routing;
  3. system does not build full inverted file;
  4. system retrieve from subcollections and combine ranked lists from each into one final retrieval list; subcollections are served by one master lexicon;
  5. can combine multiple retrieval methods.

System Summary and Timing
Organization Name: Logicon
List of Run ID's: losPA2, losPA3

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list: 580
- Controlled Vocabulary? : No
- Stemming Algorithm: No
- Term Weighting: No
- Phrase Discovery? :
- Tokenizer? : Yes
  - Patterns which are tokenized: All alphanumeric strings were tokenized. Only alphabetic tokens of length > 1 were kept. The following were then eliminated: stopwords; tokens serving as SGML tags; tokens occurring <= 2 times on each CDROM.

Statistics on Data Structures built from TREC Text

- Inverted index
- Clusters
- N-grams, Suffix arrays, Signature Files
- Knowledge Bases
  - Use of Manual Labor
- Special Routing Structures
- Other Data Structures built from TREC text
  - Run ID : losPA2, losPA3
  - Type of Structure: Corpus word-frequency table
  - Total Storage (in MB): 15
  - Total Computer Time to Build (in hours): 9
  - Automatic Process? (If not, number of manual hours): Yes
  - Brief Description of Method: Process analyzed all documents on all 3 CDROMs. Each document was tokenized, and a count was kept with each token indicating the number of documents in which it occurred.

Query construction

Automatically Built Queries (Routing)

- Topic Fields Used: None
- Average Computer Time to Build Query (in cpu seconds): CPU time was not measured. Average wall-clock time was 324 seconds.
- Method used in Query Construction
  - Terms Selected From
    - Only Documents with Relevance Judgments: Yes
  - Term Weighting with Weights Based on terms in
    - All Training Documents: Yes
    - Documents with Relevance Judgments: Yes
  - Phrase Extraction from
  - Syntactic Parsing
  - Word Sense Disambiguation using
  - Proper Noun Identification Algorithm from
  - Tokenizer
    - Patterns which are tokenized (dates, phone numbers, common patterns, etc): Same as tokenizing rules for Data Structures.
    - from Documents with Relevance Judgments: Yes
  - Heuristic Associations to Add Terms from

- Expansion of Queries using Previously-Constructed Data Structure:
- Automatic Addition of Boolean connectors or Proximity Operators using information from
  - Documents with Relevance Judgments: Yes. Any 2 of the 10 "most descriptive" tokens were specified in an AND condition for document selection. The 1000 "most descriptive" tokens were then used for document ranking.

Searching

Search Times

- Run ID : losPA2, losPA3
- Computer Time to Search (Average per Query, in CPU seconds): CPU time was not measured. Average wall-clock time was 0.45 seconds per document, or 0.009 seconds per profile per document.

Machine Searching Methods

- Boolean Matching? : Yes
- Free Text Scanning? : Yes. Search engine was the Logicon Message Dissemination System (LMDS), a COTS product.

Factors in Ranking

- Term Frequency? : Yes. Term frequency in the set of relevant training documents vs. the set of all training documents.
- Proximity of Terms? : Passage-level searching was employed as part of the scoring algorithm. A passage consisted of 100 contiguous non-stopwords, and each new passage began 50 non-stopwords after the beginning of the previous passage.
- Document Length? : Yes. Score from term weights was multiplied by the ratio of unique query tokens in document or passage to total unique tokens in document or passage.

Machine Information

- Machine Type for TREC Experiment: Sun SPARCstation 10
- Was the Machine Dedicated or Shared: Dedicated
- Amount of Hard Disk Storage (in MB): 5,000 MB
- Amount of RAM (in MB): 32 MB
- Clock Rate of CPU (in MHz): 50 MHz

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: The search engine was the Logicon Message Dissemination System (LMDS), a COTS product. The query creation and document ranking software were research prototypes integrated via its API.
- Given appropriate resources
  - Could your system run faster? : Yes
  - By how much (estimate)? : 50 percent

System Summary and Timing
Organization Name: Rutgers SCILS Interactive Track
List of Run ID's: runit1, ruibsl

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Controlled Vocabulary? : NO
- Stemming Algorithm: PORTER
  - Morphological Analysis: NO
- Term Weighting: DEFAULT INQUERY (SEE UMASS)
- Phrase Discovery? : NO
- Syntactic Parsing? : NO
- Word Sense Disambiguation? : NO
- Heuristic Associations (including short definition)? : NO
- Spelling Checking (with manual correction)? : YES
- Proper Noun Identification Algorithm? : NO
- Tokenizer? : NO
- Manually-Indexed Terms? : NO

Statistics on Data Structures built from TREC Text

- Inverted index
  - Run ID : ruint1, ruibsl
  SEE SYSTEM DESCRIPTION FOR GEORGIA TECH FOR THESE DATA
- Clusters
- N-grams, Suffix arrays, Signature Files
- Knowledge Bases
  - Use of Manual Labor
- Special Routing Structures
- Other Data Structures built from TREC text

Query construction

Automatically Built Queries (Ad-Hoc)

- Topic Fields Used: DESCRIPTION
- Method used in Query Construction
  - Term Weighting (weights based on terms in topics)? : YES
  - Phrase Extraction from Topics? : NO
  - Syntactic Parsing of Topics? : NO
  - Word Sense Disambiguation? : NO
  - Proper Noun Identification Algorithm? : NO
  - Tokenizer? : NO
  - Heuristic Associations to Add Terms? : NO
  - Expansion of Queries using Previously-Constructed Data Structure? :
    - Structure Used: NO
  - Automatic Addition of Boolean Connectors or Proximity Operators? : NO

Manually Constructed Queries (Ad-Hoc)

- Type of Query Builder
- Tools used to Build Query
  - Knowledge Base Browser? :
    - Structure Used: NO
  - Other Lexical Tools? :
- Method used in Query Construction
  - Addition of Terms not Included in Topic? :

Interactive Queries

- Initial Query Built Automatically or Manually: MANUALLY
- Type of Person doing Interaction
  - Domain Expert: NO, SEARCHERS WITH VARYING DEGREES OF EXPERIENCE
- Average Time to do Complete Interaction
  - Clock Time from Initial Construction of Query to Completion of Final Query (in minutes): 28.32
- Average Number of Iterations: 9.46
- Average Number of Documents Examined per Iteration: FULL TEXT, 2.8
- Minimum Number of Iterations: 2
- Maximum Number of Iterations: 28
- What Determines the End of an Iteration: INVOKING THE "RUN QUERY" BUTTON, OR THE "EXIT" BUTTON
- Methods used in Interaction
  - Automatic Term Reweighting from Relevant Documents? : YES
  - Automatic Query Expansion from Relevant Documents? : YES
    - All Terms in Relevant Documents added: NO
    - Only Top X Terms Added (what is X): FIVE IF ONE DOCUMENT IS RELEVANT, FIVE PLUS TWO FOR EACH SUBSEQUENT DOCUMENT IF MORE THAN ONE RELEVANT DOCUMENT
    - User Selected Terms Added: YES
  - Other Automatic Methods: NONE
  - Manual Methods
    - Using Individual Judgment (No Set Algorithm)? : YES
    - Following a Given Algorithm (Brief Description)? : NO

Searching

Machine Searching Methods

- Other: PROBABILISTIC INFERENCE NET

Factors in Ranking

- Term Frequency? : YES
- Inverse Document Frequency? : YES
- Semantic Closeness? : NO
- Position in Document? : NO
- Syntactic Clues? : NO
- Proximity of Terms? : YES
- Information Theoretic Weights? : NO
- Document Length? : YES
- Percentage of Query Terms which match? : SORT OF
- N-gram Frequency? : NO
- Word Specificity? : NO
- Word Sense Frequency? : NO
- Cluster Distance? : NO

Machine Information

- Machine Type for TREC Experiment: SUN SPARCSTATION 5
- Was the Machine Dedicated or Shared: DEDICATED
- Amount of Hard Disk Storage (in MB): 10GB
- Amount of RAM (in MB): 64MB
- Clock Rate of CPU (in MHz): 110MHZ

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: 2 PERSON-M0NTHS, FOR THE INTERFACE AND RELATED ASPECTS, ON TOP OF THE BASIC INQUERY SYSTEM

System Summary and Timing
Organization Name: University of Toronto
List of Run ID's: uofto1

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list: 24
- Controlled Vocabulary? : no
- Stemming Algorithm: no
  - Morphological Analysis: no
- Term Weighting: no
- Phrase Discovery? : yes
  - Method Used (statistical, syntactic, other): tree
- Syntactic Parsing? : no
- Word Sense Disambiguation? : no
- Heuristic Associations (including short definition)? : no
- Spelling Checking (with manual correction)? : no
- Spelling Correction? : no
- Proper Noun Identification Algorithm? : no
- Tokenizer? : no
  - Patterns which are tokenized: no
- Manually-Indexed Terms? : no
- Other Techniques for building Data Structures: tree

Statistics on Data Structures built from TREC Text

- Inverted index
  - Run ID : uofto1
  - Total Storage (in MB): 900
  - Total Computer Time to Build (in hours): 70
  - Automatic Process? (If not, number of manual hours): yes
  - Use of Term Positions? : no
  - Only Single Terms Used? : no
- Clusters
- N-grams, Suffix arrays, Signature Files
- Knowledge Bases
  - Use of Manual Labor
- Special Routing Structures
- Other Data Structures built from TREC text

Query construction

Interactive Queries

- Initial Query Built Automatically or Manually: manual
- Type of Person doing Interaction
  - Domain Expert: no
  - System Expert: no
- Average Time to do Complete Interaction
  - CPU Time (Total CPU Seconds for all Iterations): 20 seconds/query
  - Clock Time from Initial Construction of Query to Completion of Final Query (in minutes): no final query for searchers
- Average Number of Iterations: 6
- Average Number of Documents Examined per Iteration: 5
- Minimum Number of Iterations: 2
- Maximum Number of Iterations: 16

- What Determines the End of an Iteration: start of another query
- Methods used in Interaction
  - Automatic Term Reweighting from Relevant Documents? : no
  - Automatic Query Expansion from Relevant Documents? : no
    - All Terms in Relevant Documents added: no
    - Only Top X Terms Added (what is X): no
    - User Selected Terms Added: yes
  - Manual Methods
    - Using Individual Judgment (No Set Algorithm)? : yes
    - Following a Given Algorithm (Brief Description)? : no

Searching

Search Times

- Run ID : uofto1
- Computer Time to Search (Average per Query, in CPU seconds): 20
- Component Times : search 8 files

Machine Searching Methods

- Vector Space Model? : no
- Probabilistic Model? : no
- Cluster Searching? : no
- N-gram Matching? : no
- Boolean Matching? : yes
- Fuzzy Logic? : no
- Free Text Scanning? : no
- Neural Networks? : no
- Conceptual Graph Matching? : no

Machine Information

- Machine Type for TREC Experiment: 2 SUN SPARC 20/50
- Was the Machine Dedicated or Shared: 1 dedicated and 1 shared
- Amount of Hard Disk Storage (in MB): 4500 MB (dedicated)
- Amount of RAM (in MB): 32 (dedicated), and 128 (shared)
- Clock Rate of CPU (in MHz): 50

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: approximate 240 hours

System Summary and Timing
Organization Name: ETH Zurich, Switzerland
List of Run ID's: ETHI01 (interactive)

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list: 571
- Controlled Vocabulary? : no
- Stemming Algorithm: suffix stripping (Porter, 1980)
  - Morphological Analysis: no
- Term Weighting: (so-called BM25(2.0,0.0,infty,0.75)
- Phrase Discovery? : no
- Syntactic Parsing? : no
- Word Sense Disambiguation? : no
- Heuristic Associations (including short definition)? : no
- Spelling Checking (with manual correction)? : no
- Spelling Correction? : no
- Proper Noun Identification Algorithm? : no
- Tokenizer? : yes
  - Patterns which are tokenized: words
- Manually-Indexed Terms? : no
- Other Techniques for building Data Structures: no

Statistics on Data Structures built from TREC Text

- Inverted index
  - Run ID : ETHI01
  - Total Storage (in MB): 2000
  - Total Computer Time to Build (in hours): 240 for inserting all documents into the system
  - Automatic Process? (If not, number of manual hours): yes
  - Use of Term Positions? : no
  - Only Single Terms Used? : yes
- Clusters
  - Run ID : none
- N-grams, Suffix arrays, Signature Files
  - Run ID : none
- Knowledge Bases
  - Run ID : none
  - Use of Manual Labor
- Special Routing Structures
  - Run ID : none
- Other Data Structures built from TREC text
  - Run ID : ETHI01
  - Type of Structure: non-inverted files
  - Total Storage (in MB): 1500
  - Total Computer Time to Build (in hours): see above *
  - Automatic Process? (If not, number of manual hours): yes
  - Brief Description of Method: for fast processing of relevance feedback, system uses non-inverted index for updates.
- Other Data Structures built from TREC text
  - Run ID : ETHI01
  - Type of Structure: Document Info

  - Total Storage (in MB): 1100
  - Total Computer Time to Build (in hours): see above *
  - Automatic Process? (If not, number of manual hours): yes
  - Brief Description of Method: Titles of documents to show in ranked list
- Other Data Structures built from TREC text
  - Run ID : ETHI01
  - Type of Structure: Feature Numbering
  - Total Storage (in MB): 14
  - Total Computer Time to Build (in hours): see above *
  - Automatic Process? (If not, number of manual hours): yes
  - Brief Description of Method: Features are mapped to numbers
- Other Data Structures built from TREC text
  - Run ID : ETHI01
  - Type of Structure: Hidden Markov Models
  - Total Storage (in MB): 0.0005
  - Total Computer Time to Build (in hours): 2
  - Automatic Process? (If not, number of manual hours): yes
  - Brief Description of Method: Hidden Markov Models used for passage retrieval

Query construction

Automatically Built Queries (Ad-Hoc)

- Topic Fields Used: all
- Average Computer Time to Build Query (in cpu seconds): msec
- Method used in Query Construction
  - Term Weighting (weights based on terms in topics)? : feature frequency
  - Phrase Extraction from Topics? : no
  - Syntactic Parsing of Topics? : no
  - Word Sense Disambiguation? : no
  - Proper Noun Identification Algorithm? : no
  - Tokenizer? : yes
    - Patterns which are Tokenized: words
  - Heuristic Associations to Add Terms? : no
  - Expansion of Queries using Previously-Constructed Data Structure? : yes
  - Automatic Addition of Boolean Connectors or Proximity Operators? : no
  - Other: none

Automatically Built Queries (Routing)

- Average Computer Time to Build Query (in cpu seconds):
- Method used in Query Construction
  - Terms Selected From
  - Term Weighting with Weights Based on terms in
  - Phrase Extraction from
  - Syntactic Parsing
  - Word Sense Disambiguation using
  - Proper Noun Identification Algorithm from
  - Tokenizer
    - Patterns which are tokenized (dates, phone numbers, common patterns, etc): words and phrases
  - Heuristic Associations to Add Terms from
  - Expansion of Queries using Previously-Constructed Data Structure:
  - Automatic Addition of Boolean connectors or Proximity Operators using information from

Interactive Queries

- Initial Query Built Automatically or Manually: Manually
- Type of Person doing Interaction
  - Domain Expert: 2 out of 13
  - System Expert: 2 out of 13
- Average Time to do Complete Interaction
  - CPU Time (Total CPU Seconds for all Iterations): unknown
  - Clock Time from Initial Construction of Query to Completion of Final Query (in minutes): 29
- Average Number of Iterations: ???
- Minimum Number of Iterations: ???
- Maximum Number of Iterations: ???
- What Determines the End of an Iteration: user decision
- Methods used in Interaction
  - Automatic Term Reweighting from Relevant Documents? : no
  - Automatic Query Expansion from Relevant Documents? : yes
    - All Terms in Relevant Documents added: no
    - Only Top X Terms Added (what is X): 20
    - User Selected Terms Added: user selected relevant passages
  - Other Automatic Methods: none
  - Manual Methods
    - Using Individual Judgment (No Set Algorithm)? : yes
    - Following a Given Algorithm (Brief Description)? : no

Searching

- Search Times
  - Run ID : ETHI01
  - Computer Time to Search (Average per Query, in CPU seconds): 1--2 sec
  - Component Times :
- Search Times
  - Run ID : ETHI01
  - Component Times :

Machine Searching Methods

- Vector Space Model? : yes (basic method)
- Probabilistic Model? : yes (passage retrieval based on HMM)
- Cluster Searching? : no
- N-gram Matching? : no
- Boolean Matching? : no
- Fuzzy Logic? :   no
- Free Text Scanning? : no
- Neural Networks? : no
- Conceptual Graph Matching? : no
- Other: no

Factors in Ranking

- Term Frequency? : yes
- Inverse Document Frequency? : yes
- Other Term Weights? : Query Term Frequency
- Semantic Closeness? : no
- Position in Document? : no
- Syntactic Clues? : yes
- Proximity of Terms? : yes, for passage retrieval
- Information Theoretic Weights? : no

  - Document Length? : yes
  - Percentage of Query Terms which match? : no
  - N-gram Frequency? : no
  - Word Specificity? : no
  - Word Sense Frequency? : no
  - Cluster Distance? : no
  - Other: no

Machine Information

- Machine Type for TREC Experiment: SPARC Center 1000
- Was the Machine Dedicated or Shared: shared
- Amount of Hard Disk Storage (in MB): 10000
- Amount of RAM (in MB): 384
- Clock Rate of CPU (in MHz): 4x50

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: research prototype developped not exclusively for TREC, redesign of retrieval component: 0.5 person years
- Given appropriate resources
  - Could your system run faster? : yes
  - By how much (estimate)? : inserting and building the index: 10 times faster
- Features the System is Missing that would be beneficial:

Significant Areas of System

- Brief Description of features in your system which you feel impact the system and are not answered by above questions: none

System Summary and Timing
Organization Name: New York University / General Electric
List of Run ID's: nyuge1, nyuge2, nyuge3, nyuge4

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list: 365
- Controlled Vocabulary? : no
- Stemming Algorithm: yes, lexicon
  - Morphological Analysis: partial
- Term Weighting: yes
- Phrase Discovery? : yes
  - Kind of Phrase: syntactic
  - Method Used (statistical, syntactic, other): syntactic with statistical disambiguation
- Syntactic Parsing? : yes
- Word Sense Disambiguation? : no
- Heuristic Associations (including short definition)? : no
- Spelling Checking (with manual correction)? : no
- Spelling Correction? : no
- Proper Noun Identification Algorithm? : yes
- Tokenizer? : yes
  - Patterns which are tokenized: names, fixed phrases
- Manually-Indexed Terms? : no
- Other Techniques for building Data Structures: none

Statistics on Data Structures built from TREC Text

- Inverted index
  - Run ID : nyuge1 and nyuge2
  - Total Storage (in MB): 323
  - Total Computer Time to Build (in hours): 7.7
  - Automatic Process? (If not, number of manual hours): yes
  - Use of Term Positions? : no
  - Only Single Terms Used? : no
- Inverted index
  - Run ID : nyuge3 and nyuge4
  - Total Storage (in MB): 643
  - Total Computer Time to Build (in hours): 15.3
  - Automatic Process? (If not, number of manual hours): yes
  - Use of Term Positions? : no
  - Only Single Terms Used? : no
- Clusters
- N-grams, Suffix arrays, Signature Files
- Knowledge Bases
  - Use of Manual Labor
- Special Routing Structures
- Other Data Structures built from TREC text

Data Built from Sources Other than the Input Text

- Internally-built Auxiliary File
  - Use of Manual Labor
- Externally-built Auxiliary File

Query construction

Automatically Built Queries (Ad-Hoc)

- Topic Fields Used: all
- Average Computer Time to Build Query (in cpu seconds): 2
- Method used in Query Construction
  - Term Weighting (weights based on terms in topics)? : yes
  - Phrase Extraction from Topics? : yes
  - Syntactic Parsing of Topics? : yes
  - Word Sense Disambiguation? : no
  - Proper Noun Identification Algorithm? : yes
  - Tokenizer? : yes
    - Patterns which are Tokenized: names, fixed phrases
  - Heuristic Associations to Add Terms? : no
  - Expansion of Queries using Previously-Constructed Data Structure? : no
  - Automatic Addition of Boolean Connectors or Proximity Operators? : no

Automatically Built Queries (Routing)

- Topic Fields Used: desc, con
- Method used in Query Construction
  - Terms Selected From
    - Topics: yes
    - All Training Documents: no
    - Only Documents with Relevance Judgments: yes
  - Term Weighting with Weights Based on terms in
    - Topics: yes
    - All Training Documents: no
    - Documents with Relevance Judgments: yes
  - Phrase Extraction from
    - Topics: yes
    - All Training Documents: no
    - Documents with Relevance Judgments: yes
  - Syntactic Parsing
    - Topics: yes
    - All Training Documents: no
    - Documents with Relevance Judgments: yes
  - Word Sense Disambiguation using
  - Proper Noun Identification Algorithm from
    - Topics: yes
    - All Training Documents: no
    - Documents with Relevance Judgments: yes
  - Tokenizer
    - Patterns which are tokenized (dates, phone numbers, common patterns, etc): names, fixed phrases
    - from Topics: yes
    - from All Training Documents: no
    - from Documents with Relevance Judgments: yes
  - Heuristic Associations to Add Terms from
    - Topics: no
    - All Training Documents: no
    - Documents with Relevance Judgments: no
  - Expansion of Queries using Previously-Constructed Data Structure:
  - Automatic Addition of Boolean connectors or Proximity Operators using information from

Manually Constructed Queries (Ad-Hoc)

- Topic Fields Used: all
- Average Time to Build Query (in Minutes): 2
- Type of Query Builder
  - Domain Expert: no
  - Computer System Expert: yes
- Tools used to Build Query
  - Word Frequency List? : no
  - Knowledge Base Browser? : no
  - Other Lexical Tools? :
- Method used in Query Construction
  - Term Weighting? : yes
  - Boolean Connectors (AND, OR, NOT)? : no
  - Proximity Operators? : no
  - Addition of Terms not Included in Topic? : yes
    - Source of Terms: builder's imagination

Interactive Queries

- Initial Query Built Automatically or Manually: automatically
- Type of Person doing Interaction
  - Domain Expert: no
  - System Expert: yes
- Average Time to do Complete Interaction
  - Clock Time from Initial Construction of Query to Completion of Final Query (in minutes): 20
- Average Number of Iterations: 2
- Average Number of Documents Examined per Iteration: 20
- Minimum Number of Iterations: 2
- Maximum Number of Iterations: 2
- What Determines the End of an Iteration: exactly 2 iterations
- Methods used in Interaction
  - Automatic Term Reweighting from Relevant Documents? : no
  - Automatic Query Expansion from Relevant Documents? : no
    - All Terms in Relevant Documents added: no
    - Only Top X Terms Added (what is X): no
    - User Selected Terms Added: yes
  - Manual Methods
    - Using Individual Judgment (No Set Algorithm)? : yes

Searching

Search Times

- Run ID : nyuge
- Computer Time to Search (Average per Query, in CPU seconds): 60

Machine Searching Methods

- Vector Space Model? : yes
- Probabilistic Model? : no

Factors in Ranking

- Term Frequency? :yes
- Inverse Document Frequency? : yes
- Syntactic Clues? : phrases and names weighted differently
- Document Length? : yes

Machine Information

- Machine Type for TREC Experiment: Sun SparcStation 10
- Was the Machine Dedicated or Shared: dedicated
- Amount of Hard Disk Storage (in MB): 4000
- Amount of RAM (in MB): 128

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: substantial
- Given appropriate resources
  - Could your system run faster? : yes
  - By how much (estimate)? : 8-10 times
- Features the System is Missing that would be beneficial: user interface, term position information, automatic feedback

System Summary and Timing
Organization Name: University of Central Florida
List of Run ID's: UCF100 (routing run). UCFSP0 (a run on Spanish Topics 1-25).
UCFSP1 (ADHOC run on TREC-4 Spanish Topics 25-26, submitted for evaluation).
UCFSP2 (extra ADHOC run on Spanish Topics 25-26).

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Controlled Vocabulary? : Yes.
- Stemming Algorithm: No, but for the Spanish runs (UCFSP0, UCFSP1, and UCFSP2) an auxiliary program was written to "expand" infinitives into listings of other possible verb forms, and to similarly produce various forms of certain adjectives, when given in their masculine singular form.
- Phrase Discovery? : Yes, but it is a manual process when training a filter for routing runs. During training, known relevant and not-relevant documents are read and phrases that appear useful are collected. Phrases with similar meaning are then manually constructed and added to the collection. For the Spanish ADHOC runs, phrases were sometimes extracted from the document collection by "grepping" single lines containing key words. Lists of phrases with similar meaning were then constructed manually.
- Kind of Phrase: Any kind of phrase (a sequence of words) that could be useful. For example, "chief executive officer", "due to", "back to committee", "plan that would insure Americans", and "shut down trading".
- Method Used (statistical, syntactic, other): Manual observation of viewed training text.
- Word Sense Disambiguation? : Yes, but only for the Spanish run UCFSP2, where lists of words were developed whose presence would appear to indicate that other search words were being taken out of context.
- Tokenizer? :
- Manually-Indexed Terms? : Each topic has its own knowledge base which is derived from an Entity Relationship (ER) schema for the topic. For each topic, the knowledge base primarily takes the form of one or more lists (files). There are two types of files. There is a synonym file for each structure component of the ER schema, and there is a domain file for each attribute specified in the ER schema. A phrase (a sequence of words) can be an entry in a domain or synonym file. Different forms of an entry (such as carry, carries, carried,...) are also put in these files. These files are initially built from information found in a dictionary, a thesaurus, or any specialized reference source. For routing runs, training text is manually viewed to make modifications to these files. The knowledge base for a topic also includes another information (INF) file. The INF file specifies the size of a window for evaluating text, along with the importance of the individual domain and synonym files for determining relevancy of the text in the window.

Statistics on Data Structures built from TREC Text

- Inverted index
- Clusters
- N-grams, Suffix arrays, Signature Files
- Knowledge Bases
- Run ID : UCF100 (ROUTING). UCFSP0, UCFSP1, and UCFSP2 (Spanish ADHOC).
- Use of Manual Labor
- Special Routing Structures
- Run ID : UCF100. UCFSP0, UCFSP1, and UCFSP2 are Spanish ADHOC runs, but also use the data structure and method described below.
- Type of Structure: Hash table in memory to store entries in the synonym and

domain files of a particular topic filter before beginning input text scan.

- Total Storage (in MB): Insignificant.
- Total Computer Time to Build (in hours): A few seconds.
- Automatic Process? (If not, number of manual hours): Yes.
- Brief Description of Method: Before a filter starts scanning input text documents, its synonym and domain files are read and each entry is placed in a memory resident hash table.
- Other Data Structures built from TREC text

Data Built from Sources Other than the Input Text

- Internally-built Auxiliary File
- Domain (independent or specific): Domain specific, a set of synonym and domain files are built for each topic.
- Type of File (thesaurus, knowledge base, lexicon, etc.): Each file is a list of words or phrases (a sequence of words). A synonym file is constructed for a component of an ER schema. A domain file is constructed for an attribute in an ER schema. Alternate forms of words or phrases are also placed in these files.
- Total Storage (in MB): For the fifty routing topics, total storage for the synonym and domain files was 606K. For the Spanish runs (25 topics each), total storage for synonym and domain files was about 200K for the UCFSP0 run, and about 600K apiece for the UCFSP1 and UCFSP2 runs.
- Number of Concepts Represented: The concepts represented by a filter's synonym and domain files are ER schema entities, attributes, relationships, roles, subset predicates, specializations, generalizations, and categories.
- Type of Representation: An Entity Relationship (ER) Schema for a topic.
- Total Manual Time to Build (in hours): Manually building the synonym and domain files for a single topic ranged from three hours to fifty hours, the average time was twenty hours. About twenty filters were done in a rush. For about forty filters, we still feel they are not as good as they cold be if we had more time.
- Use of Manual Labor
- Mostly Manually Built using Special Interface: Yes, the files were manually built using only an editor. Initially, some files were established using reference material such as a dictionary, a thesaurus, or any specific reference book. For routing runs, the files were later modified after viewing training text. A few special interfaces were used during the training process.
- Internally-built Auxiliary File
- Domain (independent or specific): Domain specific, an information (INF) file. One is built for each topic.
- Type of File (thesaurus, knowledge base, lexicon, etc.): The INF file specifies the insertion criteria for a topic's ER schema. It represents a statement of what is relevant to the topic.
- Total Storage (in MB): Small, about 100 bytes each.
- Number of Concepts Represented: The INF file specifies the size of a sliding window (the number of words) used to determine membership in specified combinations of synonym and domain files. The importance of each synonym and domain file is also indicated in the INF file.
- Type of Representation: The insertion criteria for an Entity Relationship schema.
- Total Manual Time to Build (in hours): A few minutes to establish one, but an hour or so of wait time to see how good the INF file was for the filter. This year we tried to determine the best possible window size and best domain and synonym file weights for a filter. But, we still did not have enough time. For Spanish ADHOC runs, a few minutes to establish one, since no training is allowed.
- Use of Manual Labor
- Mostly Manually Built using Special Interface: An INF file is manually built using only an editor. For routing runs, an INF file is usually modified after

viewing training text. In an INF file, the window size and weights of individual synonym and domain files were also modified by observing successive performance evaluations over training text. This was done (when we had time) to obtain optimum performance of a filter over the training text. We did not have enough time to build optimum filters. We did not use all of the training data and we were rushed to finish for about twenty topics.

- Externally-built Auxiliary File

Query construction

Manually Constructed Queries (Ad-Hoc)

- Topic Fields Used: All.
- Average Time to Build Query (in Minutes): This is the time to sketch an ER schema for a topic (typically, a few minutes for short Spanish descriptions) plus the time to build synonym and domain files for the schema (average is ten hours) plus a few minutes to create the INF file for the topic.
- Type of Query Builder
- Domain Expert: An undergraduate Computer Science student doing independent study researched and constructed all Spanish ADHOC queries. Several of this student's Mexican-American friends were consulted for information on Mexican dialog, politics, sports, culture, etc.
- Computer System Expert: The same basic system was used for Spanish ADHOC as for routing. The undergraduate student that constructed the Spanish queries modified the system to handle Spanish.
- Tools used to Build Query
- Knowledge Base Browser? :
- Other Lexical Tools? : A lexical analyzer was used to recognize specially marked Spanish infinitives and adjectives in synonym and domain lists and expand these parts of speech to other forms.
- Method used in Query Construction
- Term Weighting? : Yes. As with our routing experiments, weight is assigned to each synonym and domain file.
- Boolean Connectors (AND, OR, NOT)? : Yes. For the Spanish ADHOC runs, the filter uses OR logic when selecting the highest weight from weights generated by a series of weight patterns. Negative weights in weight patterns implement a form of NOT. Only the UCFSP2 run uses multiple patterns and negative weights.
- Proximity Operators? : Yes. As in our routing experiments, a sliding window of user-specified size (number of words) is used.
- Addition of Terms not Included in Topic? : Yes, many!
- Source of Terms: Any kind of reference material. Some phrases were found by grepping single lines of text containing key words out of the document collection.

Manually Constructed Queries (Routing)

- Topic Fields Used: All.
- Average Time to Build Query (in Minutes): This is the time to sketch an ER schema for a topic (this should be about one hour for topic descriptions like those for Topic 001 through Topic 200) plus the time to build synonym and domain files for the schema (average time was twenty hours) plus a few minutes to create the INF file for the topic. For the fifty ROUTING queries, we started drawing ER diagrams in March. There were close to twenty students doing filters for the ROUTING topics. Explaining ER diagrams to each student was more difficult than anticipated. By late April, we were not drawing ER diagrams. The synonym and domain file concepts were still used because the students understood their purpose and it helped them decide on what to search for in a filter.
- Type of Query Builder

- Computer System Expert: The person constructing the synonym and domain files for a topic was an undergraduate student in a Computer Science independent study course.
- Tools used to Build Query
- Knowledge Base Browser? :
- Other Lexical Tools? :
- Data Used for Building Query from
- All Training Documents: No, we used training documents from just the Vol 1 and Vol 2 CDs. We did not use Vol 3 training documents. The reason we used just Vol 1 and Vol 2 documents is that not all of the ROUTING topics had training documents on Vol 3. This was a mistake because the Vol 3 training documents were probably the best, and part of the Vol 3 CD (the Ziff directory) was used for the ROUTING document collection. It would have been extremely beneficial to train on the Vol 3 Ziff directory. All of our filters for ROUTING topics in the range of Topic 051 through Topic 150 had performance at or below median performance. Topics in the range of Topic 051 through Topic 150 were the ones that had training documents in the Ziff directory of the Vol 3 CD, and we feel our filter performance was adversely affected because we did not use training documents from the Ziff directory of the Vol 3 CD.
- Documents with Relevance Judgments: Yes.
- Other Sources: Hardcopy references (such as a dictionary, a thesaurus, or a specialized reference book) were used. During training, some documents were retrieved that had no definite relevance judgment, so these documents were read and used if the student felt they were relevant.
- Method used in Query Construction
- Term Weighting? : A weight can be assigned to each synonym file and each domain file of a filter.
- Boolean Connectors (AND, OR, NOT)? : A form of AND and NOT is used when a combination of synonym and domain files is specified. A form of OR is used when different combinations of synonym and domain files are listed.
- Proximity Operators? : The sliding window (number of words) to evaluate relevancy.
- Addition of Terms not Included in Topic? : Yes!
- Source of Terms: Any kind of reference material and viewed training text.

Searching

Search Times

- Search Times
- Run ID : UCF100 (a ROUTING run).
- Computer Time to Search (Average per Query, in CPU seconds): Since each routing query was a true filter that scanned across the entire document collection, we kept track of wall clock time. To train filters across Vol 1 and Vol 2 document collections, we trained with Vol 1 in a CD drive and Vol 2 copied to a hard drive. Typically, four filters were run at once for an elapsed time of eight hours. These were runs made on a Sun SPARCserver 690MP (4 processors). We also trained filters by running them on RISC machines which accessed a hard drive copy of Vol 1 on a 386 PC running Linux and the hard drive copy of Vol 2 on the SPARCserver. Only one filter was run per RISC machine. If only one RISC machine was activated, a filter training run took nine hours. Two RISC filters took 13 hours. Three RISC filters took 18 hours to finish. For the run across the ROUTING document collection (stored in compressed form on a hard drive), five filters were run simultaneously and it took 3.5 hours for them to finish. The runs were made on the SPARCserver. The time varied depending on other use of the SPARCserver. For the Spanish ADHOC runs no record was kept of CPU time, but it generally took right at half an hour of real time to run from one to four filters simultaneously across the Spanish text using the SPARCserver, provided other

network traffic was light.
- Component Times : No component times, just run the filter across the document collection.

Machine Searching Methods

- Machine Searching Methods
- Boolean Matching? : Somewhat.
- Free Text Scanning? : Yes.
- Other: A window (number of words) to view was moved across a document collection and the window was evaluated in regard to words that satisfied the insertion criteria for an Entity Relationship (ER) schema of a topic description. This could be Conceptual Graph Matching.

Factors in Ranking

- Factors in Ranking
- Term Frequency? : Yes.
- Other Term Weights? : Yes. Each synonym or domain file can be assigned an integer "importance" determined by optimum performance over training text. We did not have enough time to determine optimum numbers.
- Position in Document? : Yes, sliding window of words to evaluate.
- Proximity of Terms? : Yes, sliding window of words to evaluate.
- Other: 1. Number of synonym and domain files for a filter. 2. Local evaluations (in the window) and a global evaluation of the entire document are used. 3. Multiple combinations of synonym and domain files are allowed for a filter.

Machine Information

- Machine Type for TREC Experiment: For training: 1. The Vol. 1 CD was copied to the hard drive of a PC running Linux (a public domain version of Unix) and functioning as an NFS node. 2. The Vol. 2 CD was copied to the hard drive of a SPARCserver 690MP (4 processors). 3. Students ran filters and viewed training text from 32 RISC 6000 machines across a network. For the UCF100 ROUTING run:
1. The ROUTING document collection was placed on the hard drive of a SPARCserver 690MP (4 processors). 2. Final filter runs were made on the SPARCserver 690MP. For UCFSP0, UCFSP1, UCFSP2 (Spanish ADHOC runs): 1. The Spanish text was copied onto the hard drive of the SPARCserver 690MP. 2. Final runs were made on the SPARCserver 690MP.
- Was the Machine Dedicated or Shared: Shared, except for the NFS node running Linux.
- Amount of Hard Disk Storage (in MB): We had access to 1000 MB on the NFS node, and 1000 MB on the SPARCserver.
- Amount of RAM (in MB): 16 MB on each of the 32 RISC 6000 machines. 16 MB on the NFS node. 128 MB on the SPARCserver.
- Clock Rate of CPU (in MHz): 33 MHz for the NFS node. Not known for the RISC 6000 machines. Not known for the SPARCserver 690MP.

System Comparisons

- Amount of "Software Engineering" which went into the Development of the Filter System during 1994: 80 hours: Purchase and install hardware and establish network access. 160 hours: Design, code, and test the basic filter scanner. 40 hours: Design, code, and test a few utilities for training. 40 hours: Figure out the rules for drawing an "atomic" ER diagram. ROUTING: 1000 hours: Establish synonym and domain files for the routing topics for the UCF100 run. Spanish ADHOC: 200 hours: Develop lex program for Spanish verb expansion. 2 hours: Modify filter to handle special Spanish characters. 40 hours: Modify filter to choose between multiple patterns, rather than summing pattern weights, and to allow for negative weighting. 500 hours: Establish ER schemas and related synonym and domain files for Spanish topics (50 in all). 40 hours: Index Spanish documents and implement utility for viewing specific documents. (Documents were read when doing the UCFSP0 run for Topic SP1 through Topic SP25.)

- Given appropriate resources
- Could your system run faster? : Yes.
- By how much (estimate)? : It is a function of how many machines are available for running a filter, and how much traffic the network will tolerate. It might be possible to put a filter on each processor of a machine like the MASPAR, and in four iterations, filter the documents on a CD in about four minutes.
- Features the System is Missing that would be beneficial: 1. A human-computer dialog interface to automate the development of an ER "atomic" schema from a person with a search request. 2. Access to electronic dictionaries, thesauri, and reference material for initial filter construction from the ER schema. 3. Utility programs to help train the filters using training documents and relevancy judgments. 4. An interface for filter modification during interactive queries.

Significant Areas of System

- Brief Description of features in your system which you feel impact the system and are not answered by above questions:

System Summary and Timing
Organization Name: CITRI, Royal Melbourne Institute of Technology
List of Run ID's: citri1, citri2, citri-sp1, citri-sp2

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list: 601
- Controlled Vocabulary? : no
- Stemming Algorithm: Lovins
  - Morphological Analysis: none
- Term Weighting: standard
- Phrase Discovery? : no
- Syntactic Parsing? : no
- Word Sense Disambiguation? : no
- Heuristic Associations (including short definition)? : no
- Spelling Checking (with manual correction)? : no
- Spelling Correction? : no
- Proper Noun Identification Algorithm? : no
- Tokenizer? : words are alphanumeric strings
- Manually-Indexed Terms? : no
- Other Techniques for building Data Structures: no

Statistics on Data Structures built from TREC Text

- Inverted index
  - Run ID : citri1, citri2
  - Total Storage (in MB): about 140 Mb
  - Total Computer Time to Build (in hours): 4
  - Automatic Process? (If not, number of manual hours): yes
  - Use of Term Positions? : no
  - Only Single Terms Used? : yes
- Clusters
- N-grams, Suffix arrays, Signature Files
- Knowledge Bases
  - Use of Manual Labor
- Special Routing Structures
- Other Data Structures built from TREC text

Query construction

Automatically Built Queries (Ad-Hoc)

- Topic Fields Used: all (ie, description)
- Average Computer Time to Build Query (in cpu seconds): 0
- Method used in Query Construction
  - Term Weighting (weights based on terms in topics)? : no
  - Phrase Extraction from Topics? :no
  - Syntactic Parsing of Topics? :no
  - Word Sense Disambiguation? :no
  - Proper Noun Identification Algorithm? :no
  - Tokenizer? : words are alphanumeric strings
  - Heuristic Associations to Add Terms? : no
  - Expansion of Queries using Previously-Constructed Data Structure? : no

  - Automatic Addition of Boolean Connectors or Proximity Operators? : no
  - Other: none

Manually Constructed Queries (Ad-Hoc)

- Topic Fields Used: description
- Average Time to Build Query (in Minutes): q
- Type of Query Builder
  - Domain Expert: no
  - Computer System Expert: yes
- Tools used to Build Query
  - Word Frequency List? : no
  - Knowledge Base Browser? : no
  - Other Lexical Tools? : no
- Method used in Query Construction
  - Term Weighting? : no
  - Boolean Connectors (AND, OR, NOT)? : no
  - Proximity Operators? : no
  - Addition of Terms not Included in Topic? : no
  - Other: none

Searching

Search Times

- Run ID : citri1/2
- Computer Time to Search (Average per Query, in CPU seconds): less than 1

Machine Searching Methods

- Vector Space Model? : yes

Factors in Ranking

- Term Frequency? : yes
- Inverse Document Frequency? : yes
- Other Term Weights? : no
- Semantic Closeness? : no
- Position in Document? : no
- Syntactic Clues? : no
- Proximity of Terms? :no
- Information Theoretic Weights? :no
- Document Length? : yes
- Percentage of Query Terms which match? : no
- N-gram Frequency? : no
- Word Specificity? : no
- Word Sense Frequency? :no
- Cluster Distance? :no
- Other: none

Machine Information

- Machine Type for TREC Experiment: Sparc 10
- Was the Machine Dedicated or Shared: shared
- Amount of Hard Disk Storage (in MB): 20 Gb
- Amount of RAM (in MB): 256
- Clock Rate of CPU (in MHz): 4 times 50

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: some
- Given appropriate resources
  - Could your system run faster? : not much
- Features the System is Missing that would be beneficial: sophistication! query processing is fairly basic

Significant Areas of System

- Brief Description of features in your system which you feel impact the system and are not answered by above questions: Compression is used throughout, to save space and improve performance. Total database size including indexes and compressed data is about 750 Mb.

System Summary and Timing
Organization Name: InTEXT Systems
List of Run ID's: INTXT2

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list: 2010
- Controlled Vocabulary? : No
- Stemming Algorithm: Modified Lovins
  - Morphological Analysis: Yes
- Term Weighting: Document by document, based on frequency and phrase length
- Phrase Discovery? : Yes
  - Kind of Phrase: Multiple word delimited by Stop Word
  - Method Used (statistical, syntactic, other): Statistical
- Syntactic Parsing? : No
- Word Sense Disambiguation? : No
- Heuristic Associations (including short definition)? : No
- Spelling Checking (with manual correction)? : No
- Spelling Correction? : No
- Proper Noun Identification Algorithm? : Yes
- Tokenizer? : Space and punctuation recognition
  - Patterns which are tokenized: Compound proper nouns
- Manually-Indexed Terms? : No
- Other Techniques for building Data Structures: Term selection based on weight and morphology using InTEXT Precision software. PreciseScope documents built with omitted wordsreplaced by noise words.

Statistics on Data Structures built from TREC Text

- Inverted index
  - Run ID : INTXT2
  - Total Storage (in MB): 600
  - Total Computer Time to Build (in hours): 80
  - Automatic Process? (If not, number of manual hours): Yes. Using InTEXT Retrieval Engine
  - Use of Term Positions? : Yes
  - Only Single Terms Used? : No
- Clusters
- N-grams, Suffix arrays, Signature Files
- Knowledge Bases
  - Use of Manual Labor
- Special Routing Structures
- Other Data Structures built from TREC text
  - Run ID : INTXT2
  - Type of Structure: PreciseScope documents
  - Total Storage (in MB): 250
  - Total Computer Time to Build (in hours): 40
  - Automatic Process? (If not, number of manual hours): Yes
  - Brief Description of Method: Each document is analysed; words and phrases are selected for indexing as outlined above, and PreciseScope documents and weighted keyword lists are created
- Other Data Structures built from TREC text
  - Run ID : INTXT2
  - Type of Structure: Weighted keywords and phrases (generated but not used

in tests)

- Total Storage (in MB): 100
- Total Computer Time to Build (in hours): Contained in PreciseScope times
- Automatic Process? (If not, number of manual hours): Yes
- Brief Description of Method: See above.

Data Built from Sources Other than the Input Text

- Internally-built Auxiliary File
  - Use of Manual Labor
- Externally-built Auxiliary File

Query construction

Automatically Built Queries (Ad-Hoc)

- Method used in Query Construction
  - Tokenizer? :
  - Expansion of Queries using Previously-Constructed Data Structure?:

Automatically Built Queries (Routing)

- Method used in Query Construction
  - Terms Selected From
  - Term Weighting with Weights Based on terms in
  - Phrase Extraction from
  - Syntactic Parsing
  - Word Sense Disambiguation using
  - Proper Noun Identification Algorithm from
  - Tokenizer
  - Heuristic Associations to Add Terms from
  - Expansion of Queries using Previously-Constructed Data Structure:
  - Automatic Addition of Boolean connectors or Proximity Operators using information from

Manually Constructed Queries (Ad-Hoc)

- Topic Fields Used: None
- Average Time to Build Query (in Minutes): 30. Very variable. (Mainly due to lack of knowledge of US current affairs)
- Type of Query Builder
  - Computer System Expert: Yes
- Tools used to Build Query
  - Word Frequency List? : No
  - Knowledge Base Browser? :
  - Other Lexical Tools? :
- Method used in Query Construction
  - Term Weighting? : Yes. Minimal. Usually one group of alternative terms was manadatory
  - Boolean Connectors (AND, OR, NOT)? : AND, OR for mandatory terms.
  - Proximity Operators? : Yes. Phrase and same paragraph
  - Addition of Terms not Included in Topic? : Yes
    - Source of Terms: General knowledge and research

Manually Constructed Queries (Routing)

- Type of Query Builder
- Tools used to Build Query

- Knowledge Base Browser? :
  - Other Lexical Tools? :
- Data Used for Building Query from
- Method used in Query Construction
  - Addition of Terms not Included in Topic? :

Interactive Queries

- Type of Person doing Interaction
- Average Time to do Complete Interaction
- Methods used in Interaction
  - Automatic Query Expansion from Relevant Documents? :
  - Manual Methods

Searching

Search Times

- Run ID : INTXT2
- Computer Time to Search (Average per Query, in CPU seconds): 120

Machine Searching Methods

- Machine Searching Methods
  - Boolean Matching? : Yes, in part

Factors in Ranking

- Factors in Ranking
  - Term Frequency? : Yes
  - Inverse Document Frequency? : No
  - Other Term Weights? : Generated automatically by determining their discriminating power in query or defined manually.
  - Position in Document? : Yes
  - Proximity of Terms? : Yes. Phrases and paragraphs
  - Percentage of Query Terms which match? : Yes

Machine Information

- Machine Type for TREC Experiment: Pentium
- Was the Machine Dedicated or Shared: Yes
- Amount of Hard Disk Storage (in MB): 4000
- Amount of RAM (in MB): 16
- Clock Rate of CPU (in MHz): 90

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: Both InTEXT Retrieval Engine and InTEXT Precision are commercial products based on an elapsed development time of 10+years. Precise figures not available.
- Given appropriate resources
  - Could your system run faster? : Yes. Elimination of intermediate files, multi-threading
  - By how much (estimate)? : 5-10

System Summary and Timing
Organization Name: MulitText Project, Department of Computer Science,
University of Waterloo
List of Run ID's: uwgcl1 (ID used for both adhoc and routing)

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list: 0 (no stopword list was used)
- Controlled Vocabulary? : no
- Stemming Algorithm: none
- Term Weighting: none
- Phrase Discovery? :
- Tokenizer? : yes
  - Patterns which are tokenized: Basic token is a sequences of alphanumeric characters with alphabetic characters mapped to lower case. Tokens with length greater than one consisting entirely of upper case alphabetic characters were doubly indexed in both upper and lower case. SGML tags are recognized and indexed as such.

Statistics on Data Structures built from TREC Text

- Inverted index
  - Run ID : uwgcl1
  - Total Storage (in MB): aproximately 60% of text size
  - Total Computer Time to Build (in hours): 32 hours (adhoc); 14 hours (routing)
  - Automatic Process? (If not, number of manual hours): yes
  - Use of Term Positions? : yes
  - Only Single Terms Used? : yes
- Clusters
- N-grams, Suffix arrays, Signature Files
- Knowledge Bases
  - Use of Manual Labor
- Special Routing Structures
- Other Data Structures built from TREC text

Query construction

Manually Constructed Queries (Ad-Hoc)

- Topic Fields Used: description
- Average Time to Build Query (in Minutes): 30 minutes
- Type of Query Builder
  - Domain Expert: no
  - Computer System Expert: yes
- Tools used to Build Query
  - Knowledge Base Browser? :
  - Other Lexical Tools? :
- Method used in Query Construction
  - Term Weighting? : no
  - Boolean Connectors (AND, OR, NOT)? : yes
  - Proximity Operators? : yes
  - Addition of Terms not Included in Topic? : yes
    - Source of Terms: personal knowledge

Manually Constructed Queries (Routing)

- Topic Fields Used: title, description, narrative, nationality, concepts, definitions
- Average Time to Build Query (in Minutes): 30 minutes
- Type of Query Builder
  - Domain Expert: no
  - Computer System Expert: yes
- Tools used to Build Query
  - Knowledge Base Browser? :
  - Other Lexical Tools? :
- Data Used for Building Query from
  - Documents with Relevance Judgments: very limited use
- Method used in Query Construction
  - Term Weighting? : no
  - Boolean Connectors (AND, OR, NOT)? : yes
  - Proximity Operators? : yes
  - Addition of Terms not Included in Topic? : yes
  - Other: GCL containment and ordering operators

Searching

Search Times

- Run ID : uwgcl1
- Computer Time to Search (Average per Query, in CPU seconds): 40 seconds (adhoc, elapsed) 10 seconds (routing, elapsed)
- Component Times : Each query was composed of several sub-queries, each of which was run separately. An average of 1.9 sub-queries per query for routing run gives an average search time of 5 seconds per sub-query. An average of 2.2 sub-queries per query for ad-hoc run gives an average search time of 18 seconds per sub-query.

Machine Searching Methods

- Boolean Matching? : yes
- Other: GCL query matching (see main paper)

Factors in Ranking

- Term Frequency? : yes
- Proximity of Terms? : yes
- Other: solution density (see main paper for explanation)

Machine Information

- Machine Type for TREC Experiment: DEC Alpha 2000/300
- Was the Machine Dedicated or Shared: Dedicated
- Amount of Hard Disk Storage (in MB): 10GB
- Amount of RAM (in MB): 64MB
- Clock Rate of CPU (in MHz): 150MHz

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: Base retrieval system is a research prototype. Approximately two weeks of software development was specific to TREC-4.
- Given appropriate resources
  - Could your system run faster? : yes
  - By how much (estimate)? : factor of two

System Summary and Timing
Organization Name: CLARITECH Corporation
List of Run ID's: CLARTF, CLARTN

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list: N/A
- Controlled Vocabulary? : N/A
- Stemming Algorithm: N/A
  - Morphological Analysis: A comprehensive inflectional morphology is used to produce word roots. Participles are retained in surface forms. (Although normalization is possible.) No derivational morphology is used.
- Term Weighting: 1) TF*IDF over phrases is used for retrieval
  2) An importance coefficient is assigned to TF*IDF for query terms
  3) a combination of statistics, linguistic-structure analysis, and heuristics is used for thesaurus extraction; the statistical measures include frequency and distribution.
- Phrase Discovery? : Yes.
  - Kind of Phrase: simplex noun phrases -- not including post-nominal appositive, prepositional, or participial phrases or relative clauses
  - Method Used (statistical, syntactic, other): A deterministic, rule-based parser nominates linguistic-constituent structure; a filter retains only simplex noun phrases for indexing purposes.
- Syntactic Parsing? : Yes. (see above)
- Word Sense Disambiguation? : The parser grammar includes heuristics for syntactic category disambiguation.
- Heuristic Associations (including short definition)? : No.
- Spelling Checking (with manual correction)? : No.
- Spelling Correction? : No.
- Proper Noun Identification Algorithm? : Words not identified in the lexicon (about 100,000 root forms of English) are assumed to be "candidate proper nouns". The grammar accommodates structure that includes proper names; this technique does not require case-sensitive clues (e.g., capitalization) to be effective. etc.
- Tokenizer? : None.
- Manually-Indexed Terms? : No.
- Other Techniques for building Data Structures:
  1) Thesaurus discovery -- which we use for query-vector augmentation -- involves the identification of core characteristic terminology over a document set. The process ranks all terms according to scores along several parameters and then selects the subset of terminology that optimizes the scores. 2) Document Windows -- documents are decomposed into overlapping windows of 50 terms each. These windows are applied in thesaurus extraction and as a component in the document similarity calculation.

Statistics on Data Structures built from TREC Text

- Inverted index In the version of the system configured for experimentation, we do not build an inverted index for the whole corpus, but simply index the formal query structures with expanded statistical information from the full corpus. This allows us to quickly change parameters at all levels of the system.
  - Run ID : CLARTF
  - Total Storage (in MB): 5.1

  - Total Computer Time to Build (in hours): 10.5 (It takes approximately 8.75 minutes to invert the full set of 50 query vectors and merge in the global corpus statistics. The figure of 10.5 hours represents the time required to collect term distribution statistics for the entire corpus, but this only needs to be performed once for all of the runs.)
  - Automatic Process? (If not, number of manual hours): Yes.
  - Use of Term Positions? : No.
  - Only Single Terms Used? : all terms used.
  - Run ID : CLARTN
  - Total Storage (in MB): 5.1
  - Total Computer Time to Build (in hours): 10.5 (For the CLARTN run, it took approximately 9.25 minutes to process the query vectors. For more information on index creation, see above.)
  - Automatic Process? (If not, number of manual hours): Yes.
  - Use of Term Positions? : No.
  - Only Single Terms Used? : all terms used.
- Clusters None.
- N-grams, Suffix arrays, Signature Files None.
- Knowledge Bases None.
  - Use of Manual Labor
- Special Routing Structures N/A
- Other Data Structures built from TREC text
  - Run ID : CLARTF
  - Type of Structure: Automatic Feedback Thesaurus
  - Total Storage (in MB): 0.14
  - Total Computer Time to Build (in hours): 4.51 (This includes 4.33 hours to perform an initial retrieval pass, and .19 hours to construct thesauri from those results. Note that the initial retrieval pass has to be executed only once for both the feedback and distractor thesauri.)
- Other Data Structures built from TREC text
  - Run ID : CLARTF and CLARTN
  - Type of Structure: Automatic Distractor Thesaurus
  - Total Storage (in MB): 2.7
  - Total Computer Time to Build (in hours): 6.97 (This includes 4.33 hours to peform the initial retrieval pass and 2.64 hours for extraction of the thesauri. See Automatic Feedback Thesaurus, above. Since the initial query vectors from CLARTN and CLARTF are identical, the two separate runs are able to use the same distractor thesauri.)
  - Brief Description of Method: A very large, first-order thesaurus is constructed based on a large sample of relatively high-scoring, but probably non-relevant document windows. (The discovered terms are regarded as "found distractors.") For the purpose of TREC 4, we used a sample of 500 windows from ranks 37,500 - 39,000 based on the initial retrieval over the corpus. The thesaurus was extracted at the 70 percent threshold.
- Other Data Structures built from TREC text
  - Run ID : CLARTN
  - Type of Structure: Automatic Feedback Thesaurus
  - Total Storage (in MB): 0.17
  - Total Computer Time to Build (in hours): 4.66 (This includes 4.33 hours to perform an initial retrieval pass, and .33 hours to construct thesauri from those results. Note that the initial retrieval pass from the CLARTF pass is re-used here.)
  - Automatic Process? (If not, number of manual hours): Yes.
  - Brief Description of Method: A thesaurus is constructed as described for the CLARTF run, but no required terms filter is applied. The top-scoring 50 document windows are submitted directly to thesaurus extraction.

- Other Data Structures built from TREC text
  - Run ID : CLARTN and CLARTF
  - Type of Structure: Sampled Distractor Terms
  - Total Storage (in MB): 0.15
  - Total Computer Time to Build (in hours): 0.01
  - Automatic Process? (If not, number of manual hours): Yes.
  - Brief Description of Method: The top-2,000 most-frequent words from the entire retrieval corpus were taken as a representative sample for purposes of initial, ad-hoc querying.

Data Built from Sources Other than the Input Text

- Internally-built Auxiliary File
  - Type of File (thesaurus, knowledge base, lexicon, etc.): lexicon
  - Total Storage (in MB): 1.75
  - Number of Concepts Represented: 90,057 root forms
  - Type of Representation: root/syntactic-category pairs
  - Total Computer Time to Build (in hours): N/A
  - Total Manual Time to Build (in hours): The CLARIT lexicon was manually constructed using word-lists extracted from on-line sources during early phases of the CLARIT research project. (1988--89)
  - Total Manual Time to Modify for TREC (if already built): No modification was required.
  - Use of Manual Labor
    - Mostly Manually Built using Special Interface: Yes.
- Externally-built Auxiliary File

Query construction

Manually Constructed Queries (Ad-Hoc)

- Topic Fields Used: title and description
- Average Time to Build Query (in Minutes): 15 min./query
- Type of Query Builder
  - Domain Expert: No.
  - Computer System Expert: Yes.
- Tools used to Build Query
  - Word Frequency List? : No.
  - Knowledge Base Browser? : No.
  - Other Lexical Tools? : No.
  - CLARIT Retrieval System : Candidate query vectors are suggested by automatic query construction using CLARIT parsing; these vectors are manually refined by the user. Initial query performance is evaluated on other databases, including data from TREC disk 1.
- Method used in Query Construction
  - Term Weighting? : Topic terms are weighted using TF*IDF * Importance, where the Importance coefficient for query source terminology is assigned manually.

Searching

Search Times

- Computer Time to Search (Average per Query, in CPU seconds): The querying program processed all 50 topics (either routing or ad-hoc) in parallel on five machines, without the use of any inverted index or postings file. Also, this simple prototype architecture required two passes: one for full documents and one for document windows.

Processing took approximately 6 hours.

Machine Searching Methods

- Vector Space Model? : Yes -- using a cosine distance measure. Furthermore, the vector space does not fix the document vector length component of the cosine distance formula. Rather, the length of any document vector is allowed to vary depending on the terms present in the query; only the terms in the query vector are considered to be 'active' for any given distance calculation, and all other terms in the document are ignored. Since query vectors may include zero-coefficient distractor terms, each query can specify its own list of 'active terminology', thereby allowing query-specific control of the document representation.
- Other: Document windows used for feedback in the CLARTF run are subjected to a regular expression search for required terms, as described above.

Factors in Ranking

- Term Frequency? : TF_AUG = (0.5 + 0.5 * TF / MAX_TF)
- Inverse Document Frequency? : IDF = Log_2(Number of docs in the corpus / Number of docs containing term) + 1
- Other Term Weights? : An importance coefficient is manually assigned for initial query terms. (see above)
- Semantic Closeness? : No.
- Position in Document? : No.
- Syntactic Clues? : No.
- Proximity of Terms? : No.
- Information Theoretic Weights? : No.
- Document Length? : Only as this is implicitly captured in the cosine distance measure.
- Percentage of Query Terms which match? : Only as this is implicitly captured in the cosine distance measure.
- N-gram Frequency? : No.
- Word Specificity? : No.
- Word Sense Frequency? : No.
- Cluster Distance? : No.
- Other: The final score for a document is computed as the weighted average of the score for the highest scoring window in the document, and the score for the complete document vector: score = ((20 * document_score + max_window_score) / 21)

Machine Information

- Machine Type for TREC Experiment: 4 x DEC 3000/400, 1 x DEC 3000/600 (Alpha/AXP) running DEC OSF/1
- Was the Machine Dedicated or Shared: Dedicated
- Amount of Hard Disk Storage (in MB): 15,000
- Amount of RAM (in MB): 3 @ 128 meg, 2 @ 64 meg
- Clock Rate of CPU (in MHz): 4 @ 133.33 MHz, 1 @ 175 MHz

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: The system used for TREC-4 processing was configured for flexibility, rather than speed. Essentially, it is designed to store almost all intermediate results and allow for new modules to be added in a dynamic fashion.

- Given appropriate resources
  - Could your system run faster? : Yes, by simply pinning down a set of configuration options and using the normal CLARIT retrieval modules.
  - By how much (estimate)? : The standard CLARIT system indexes at a rate of 100 megabytes/hour (on a 100-MHz processor); queries of 30-50 terms over 1-2 gigabytes of data typically require < 5 secs. of cpu.
- Features the System is Missing that would be beneficial: The CLARIT-TREC-4 System did not take advantage of several processing options that may have given improved results, including feedback term re-weighting, tokenization, sub-lexicon discovery over trainings sets, and EQ-class discovery for thesaural terms.

System Summary and Timing

Organization Name: National Security Agency

List of Run ID's: ACQADH (ad-hoc), ACQROU (routing), ACQUNC (uncorrupted) ACQC10 (10% corruption) ACQC20 (20% corruption), ACQINT (interactive), ACQSPA (Spanish), ACQHPr (high precision filtering), ACQHRe (high recall filtering) ACQMID (mid-performance filtering)

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list: 0
- Controlled Vocabulary? : none
- Stemming Algorithm:
- Term Weighting: yes
- Phrase Discovery? :
- Tokenizer? :

Statistics on Data Structures built from TREC Text

- Inverted index
  - Run ID : ACQADH, ACQUNC, ACQC10, ACQC20, ACQINT, ACQSPA, ACQHPr, ACQHRe, ACQMID, ACQROU
  - Total Storage (in MB): less than 10
  - Total Computer Time to Build (in hours): 0.1 to 0.2
  - Automatic Process? (If not, number of manual hours): yes
  - Use of Term Positions? : no
  - Only Single Terms Used? : yes
- Clusters
- N-grams, Suffix arrays, Signature Files
  - Run ID : ACQADH, ACQUNC, ACQC10, ACQC20, ACQINT, ACQSPA, ACQHPr, ACQHRe, ACQMID, ACQROU
  - Total Storage (in MB): less than 10
  - Total Computer Time to Build (in hours): less than 2
  - Automatic Process? (If not, number of manual hours):yes
  - Brief Description of Method: the n-grams in each document were tallied
- Knowledge Bases
  - Use of Manual Labor
- Special Routing Structures
- Other Data Structures built from TREC text

Query construction

Automatically Built Queries (Routing)

- Topic Fields Used: none
- Average Computer Time to Build Query (in cpu seconds): 10
- Method used in Query Construction
  - Terms Selected From
    - Only Documents with Relevance Judgments: yes
  - Term Weighting with Weights Based on terms in
    - Documents with Relevance Judgments: yes
  - Phrase Extraction from
  - Syntactic Parsing
  - Word Sense Disambiguation using
  - Proper Noun Identification Algorithm from
  - Tokenizer
  - Heuristic Associations to Add Terms from

- Expansion of Queries using Previously-Constructed Data Structure:
- Automatic Addition of Boolean connectors or Proximity Operators using information from

Manually Constructed Queries (Ad-Hoc)

- Topic Fields Used: description
- Average Time to Build Query (in Minutes): 2 minutes
- Type of Query Builder
  - Computer System Expert: yes
- Tools used to Build Query
  - Knowledge Base Browser? :
  - Other Lexical Tools? :
- Method used in Query Construction
  - Addition of Terms not Included in Topic? : yes
    - Source of Terms: general knowledge of user

Interactive Queries

- Initial Query Built Automatically or Manually: manually, as with adhoc
- Type of Person doing Interaction
  - System Expert: yes
- Average Time to do Complete Interaction
  - CPU Time (Total CPU Seconds for all Iterations): less than 10
  - Clock Time from Initial Construction of Query to Completion of Final Query (in minutes): less than 1
- Methods used in Interaction
  - Automatic Query Expansion from Relevant Documents? :
  - Other Automatic Methods: ACQUAINTANCE returned the 1000 top scoring documents from the database for each topic. These scores and documents were handed to the PARENTAGE information visualization system, which arranged the documents into clusters of most related documents, and labelled each cluster with the words and phrases which best characterized the significant elements of that cluster. By rapidly scanning the labels on the clusters, those groups of documents most closely matching the topic description were identified.
  - Manual Methods

Searching

Search Times

- Run ID : ACQADH, ACQUNC, ACQC10, ACQC20, ACQSPA, ACQHPr, ACQHRe, ACQMID, ACQROU
- Computer Time to Search (Average per Query, in CPU seconds): roughly 30 seconds per query

Machine Searching Methods

- Vector Space Model? : yes
- Probabilistic Model? : yes
- N-gram Matching? : yes
- Other: n-gram frequencies are offset according to population means

Factors in Ranking

- Term Frequency? : yes, where terms are n-grams
- N-gram Frequency? : yes

Machine Information

- Machine Type for TREC Experiment: CRAY C 90
- Was the Machine Dedicated or Shared: heavily shared
- Amount of Hard Disk Storage (in MB): 4,000
- Amount of RAM (in MB): 128
- Clock Rate of CPU (in MHz): 250

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: one-half man year
- Given appropriate resources
  - Could your system run faster? : yes
  - By how much (estimate)? : one to three orders of magnitude with improved algorithm design and hardware implementation

System Summary and Timing
Organization Name: Dublin City UNiversity
List of Run ID's: DCU951 DCU952

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list: 676.
- Controlled Vocabulary? : Yes.
- Stemming Algorithm: WordNet Stemmer.
  - Morphological Analysis: Yes.
- Term Weighting: Yes.
- Phrase Discovery? :
  - Kind of Phrase: Co-Locations.
  - Method Used (statistical, syntactic, other): Other.
- Syntactic Parsing? : No.
- Word Sense Disambiguation? : No.
- Spelling Checking (with manual correction)? : No.
- Spelling Correction? : No.
- Proper Noun Identification Algorithm? : No.
- Tokenizer? : No.
- Manually-Indexed Terms? : No.
- Other Techniques for building Data Structures: No.

Statistics on Data Structures built from TREC Text

- Inverted index
  - Total Storage (in MB): 253
  - Total Computer Time to Build (in hours): 4.5
  - Automatic Process? (If not, number of manual hours): Yes.
  - Use of Term Positions? : No.
  - Only Single Terms Used? : No.

Query construction

Automatically Built Queries (Ad-Hoc)

- Topic Fields Used: DESC
- Average Computer Time to Build Query (in cpu seconds): < 1
- Method used in Query Construction
  - Term Weighting (weights based on terms in topics)? : Yes
  - Phrase Extraction from Topics? : Yes
  - Syntactic Parsing of Topics? : No
  - Word Sense Disambiguation? : DCU951 Yes, DCU952 No.
  - Proper Noun Identification Algorithm? : No
  - Tokenizer? : No
  - Heuristic Associations to Add Terms? : No.
  - Expansion of Queries using Previously-Constructed Data Structure? : DCU951 Yes, DCU952 No.
  - Automatic Addition of Boolean Connectors or Proximity Operators? : No.

Searching

Search Times

- Computer Time to Search (Average per Query, in CPU seconds): 10

Machine Searching Methods

- Vector Space Model? : Yes

Factors in Ranking

- Term Frequency? : Yes
- Inverse Document Frequency? : Yes

Machine Information

- Machine Type for TREC Experiment: Sparc 20
- Was the Machine Dedicated or Shared: Shared
- Amount of Hard Disk Storage (in MB): 1200
- Amount of RAM (in MB): 96

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: 4 Months
- Given appropriate resources
  - Could your system run faster? : Yes
  - By how much (estimate)? : 30% to 40%
- Features the System is Missing that would be beneficial: Index Compression

Significant Areas of System

- Brief Description of features in your system which you feel impact the system and are not answered by above questions:
  Query Space Processing approach

System Summary and Timing
Organization Name: FS Consulting
List of Run ID's: fsclt1, fsclt2

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list: 626, an initial list of 377 stop words are used and added to during the indexing process
- Controlled Vocabulary? : no
- Stemming Algorithm: plural stemmer (though one can select not to stem or to use the Porter stemmer as alternatives)
  - Morphological Analysis: no
- Term Weighting: applied at search time, tf.idf
- Phrase Discovery? : no
- Syntactic Parsing? : no
- Word Sense Disambiguation? : no
- Heuristic Associations (including short definition)? : no
- Spelling Checking (with manual correction)? : no
- Spelling Correction? : no
- Proper Noun Identification Algorithm? : no
- Tokenizer? : no
- Manually-Indexed Terms? : no
- Other Techniques for building Data Structures: no

Statistics on Data Structures built from TREC Text

- Inverted index
  - Run ID : all runs
  - Total Storage (in MB): 800
  - Total Computer Time to Build (in hours): 14
  - Automatic Process? (If not, number of manual hours): yes
  - Use of Term Positions? : yes
  - Only Single Terms Used? : no
- Clusters
- N-grams, Suffix arrays, Signature Files
- Knowledge Bases
  - Use of Manual Labor
- Special Routing Structures
- Other Data Structures built from TREC text

Query construction

Automatically Built Queries (Ad-Hoc)

- Topic Fields Used: no applicable
- Average Computer Time to Build Query (in cpu seconds): approx. 1 second per query
- Method used in Query Construction
  - Term Weighting (weights based on terms in topics)? : yes, see 'other' section below
  - Phrase Extraction from Topics? : no
  - Syntactic Parsing of Topics? : no
  - Word Sense Disambiguation? : no
  - Proper Noun Identification Algorithm? : no
  - Tokenizer? : no
  - Heuristic Associations to Add Terms? : no

- Expansion of Queries using Previously-Constructed Data Structure? :
  - Automatic Addition of Boolean Connectors or Proximity Operators? : no
  - Other: Automatic queries were built using the result set generated from the manually built queries (ad-hoc). The ten most important words (using tf.idf) were extracted from the top three documents (the documents were combined) and added as relevance feedback terms to the original queries. While this did not add additional documents to the result set, the set was reranked.

Manually Constructed Queries (Ad-Hoc)

- Topic Fields Used: Description
- Average Time to Build Query (in Minutes): 2
- Type of Query Builder
  - Domain Expert: no, we simulated an 'average' or 'naive' searcher
  - Computer System Expert: no
- Tools used to Build Query
  - Word Frequency List? : no, but a stop word list was consulted
  - Knowledge Base Browser? : no
  - Other Lexical Tools? : no
- Method used in Query Construction
  - Term Weighting? : no
  - Boolean Connectors (AND, OR, NOT)? : yes
  - Proximity Operators? : yes
  - Addition of Terms not Included in Topic? : yes
    - Source of Terms: the user searcher allowed to expand terms present in the topic if it made sense, for example 'nanny' was added as a search term for a topic that was about 'au-pairs'. One could also use wildcards in the search terms or use soundex on the search terms.

Searching

Search Times

- Run ID : fsclt1
- Computer Time to Search (Average per Query, in CPU seconds): 18 seconds
- Component Times : searching & ranking time: 5 seconds, downloading and saving of results by client: 13 seconds (note these are 'real' times, total run of 50 question took 900 seconds and include server startup time, search & ranking, transmission of results to the client and saving of these results to a file)
- Run ID : fsclt2
- Computer Time to Search (Average per Query, in CPU seconds): 36 seconds
- Component Times : searching & ranking time: 23 seconds, downloading and saving of results by client: 13 seconds (note these are 'real' times, total run of 50 question took 1800 seconds and include server startup time, search & ranking, transmission of results to the client and saving of these results to a file)

Machine Searching Methods

- Vector Space Model? : no
- Probabilistic Model? : yes
- Cluster Searching? : no
- N-gram Matching? : no
- Boolean Matching? : yes
- Fuzzy Logic? : no
- Free Text Scanning? : no

- Neural Networks? : no
- Conceptual Graph Matching? : no
- Other: phrase, soundex and wildcard searching are all possible

Factors in Ranking

- Term Frequency? : yes
- Inverse Document Frequency? : yes
- Other Term Weights? : no
- Semantic Closeness? : no
- Position in Document? : no
- Syntactic Clues? : no
- Proximity of Terms? : no
- Information Theoretic Weights? : no
- Document Length? : yes
- Percentage of Query Terms which match? : yes
- N-gram Frequency? : no
- Word Specificity? : no
- Word Sense Frequency? : no
- Cluster Distance? : no

Machine Information

- Machine Type for TREC Experiment: SparcStation 5
- Was the Machine Dedicated or Shared: dedicated
- Amount of Hard Disk Storage (in MB): 4GB
- Amount of RAM (in MB): 32
- Clock Rate of CPU (in MHz): 70

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: two person/years
- Given appropriate resources
  - Could your system run faster? : yes
  - By how much (estimate)? : Linked to the speed of the hardware and the amount of memory available for indexing/searching

System Summary and Timing
Organization Name: George Mason University
List of Run ID's: English: gmu1 (manual), gmu2 (automatic)
Corrupted: gmuc0, gmu10 Spanish: gmumanual, gmuauto

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list: 144
- Controlled Vocabulary? : No
- Stemming Algorithm: None
  - Morphological Analysis: No
- Term Weighting: Yes, tf-idf
- Phrase Discovery? : yes
  - Kind of Phrase: two adjacent terms, not separated by stop terms or punctuation
  - Method Used (statistical, syntactic, other): syntactic
- Syntactic Parsing? : no
- Word Sense Disambiguation? : no
- Heuristic Associations (including short definition)? : no
- Spelling Checking (with manual correction)? :no
- Spelling Correction? : no
- Proper Noun Identification Algorithm? : no
- Tokenizer? : no
- Manually-Indexed Terms? : no

Statistics on Data Structures built from TREC Text

- Inverted index
  - Run ID : gmu1, gmu2
  - Total Storage (in MB): 248.3
  - Total Computer Time to Build (in hours): 2:52:15
  - Automatic Process? (If not, number of manual hours): yes
  - Use of Term Positions? : no
  - Only Single Terms Used? : no
- Clusters
- N-grams, Suffix arrays, Signature Files
  - Run ID : gmuc0, gmuc10, gmuman, gmuauto
  - Automatic Process? (If not, number of manual hours): no
  - Brief Description of Method: For corrupted data, 4-grams were used with automatic query reduction based on term frequency across the entire document collection. For Spanish, 5-grams were used with no query reduction.

Query construction

Automatically Built Queries (Ad-Hoc)

- Topic Fields Used: all
- Average Computer Time to Build Query (in cpu seconds): 60
- Method used in Query Construction
  - Term Weighting (weights based on terms in topics)? : yes
  - Phrase Extraction from Topics? : yes
  - Syntactic Parsing of Topics? : no
  - Word Sense Disambiguation? : no
  - Proper Noun Identification Algorithm? : no
  - Tokenizer? :

    - Patterns which are Tokenized: no
  - Heuristic Associations to Add Terms? : no
    - Structure Used: none
  - Automatic Addition of Boolean Connectors or Proximity Operators? : no

Manually Constructed Queries (Ad-Hoc)

- Topic Fields Used: all
- Average Time to Build Query (in Minutes): 10
- Type of Query Builder
  - Domain Expert: no
  - Computer System Expert: yes
- Tools used to Build Query
  - Word Frequency List? : no
  - Knowledge Base Browser? :
    - Structure Used: no
  - Other Lexical Tools? :
- Method used in Query Construction
  - Term Weighting? : yes
  - Boolean Connectors (AND, OR, NOT)? : yes
  - Proximity Operators? : no
  - Addition of Terms not Included in Topic? : yes
    - Source of Terms: none

Searching

Search Times

- Run ID : gmu2
- Computer Time to Search (Average per Query, in CPU seconds): approximately 60

Machine Searching Methods

- Vector Space Model? : yes
- N-gram Matching? : yes
- Boolean Matching? :yes

Factors in Ranking

- Term Frequency? : yes
- Inverse Document Frequency? : yes
- Document Length? : yes (normalization)
- Percentage of Query Terms which match? : no
- N-gram Frequency? : yes
- Word Specificity? : no
- Word Sense Frequency? : no

Machine Information: gmuc0, gmuc10, gmu1, gmuauto

- Machine Type for TREC Experiment: SUN Sparc 2000, 18 processor
- Was the Machine Dedicated or Shared: dedicated
- Amount of Hard Disk Storage (in MB): 54000
- Amount of RAM (in MB): 2000

Machine Information: gmu1, gmuman

- Machine Type for TREC Experiment: AT&T DBC-1012 Database Machine

- Was the Machine Dedicated or Shared: dedicated
- Amount of Hard Disk Storage (in MB): 25000

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: 1 person year for IR system, 1 person year for Relational
- Given appropriate resources
  - Could your system run faster? : yes
  - By how much (estimate)? : IR, 50 percent, relational 25
- Features the System is Missing that would be beneficial: IR prototype currently is not parallelized, both lack passage based retrieval and relevance feedback.

System Summary and Timing
Organization Name: Information Technology Institute, Singapore
List of Run ID's: itidp1, itidp2

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list: 429
- Controlled Vocabulary? : 0
- Stemming Algorithm: Porter's
- Term Weighting: idf
- Phrase Discovery? : No
- Syntactic Parsing? : No
- Word Sense Disambiguation? : No
- Heuristic Associations (including short definition)? : No
- Spelling Checking (with manual correction)? : No
- Spelling Correction? : No
- Proper Noun Identification Algorithm? : No
- Tokenizer? : No
- Manually-Indexed Terms? : No
- Other Techniques for building Data Structures: No

Statistics on Data Structures built from TREC Text

- Inverted index
  - Run ID : itidp1, itidp2
  - Total Storage (in MB): 900 (only rel docs and test data)
  - Total Computer Time to Build (in hours): 100
  - Automatic Process? (If not, number of manual hours): yes
  - Use of Term Positions? : no
  - Only Single Terms Used? : yes
- Clusters
- N-grams, Suffix arrays, Signature Files
- Knowledge Bases
  - Use of Manual Labor
- Special Routing Structures
- Other Data Structures built from TREC text

Query construction

Automatically Built Queries (Routing)

- Topic Fields Used: Used training documents only
- Average Computer Time to Build Query (in cpu seconds): 1300
- Method used in Query Construction
  - Terms Selected From
    - Topics: no
    - All Training Documents: yes
  - Term Weighting with Weights Based on terms in
    - Topics: no
    - All Training Documents: yes

Searching

Search Times

- Run ID : itidp1, itidp2
- Computer Time to Search (Average per Query, in CPU seconds): 300

Machine Searching Methods

- Vector Space Model? : yes
- Boolean Matching? : yes

Factors in Ranking

- Term Frequency? : yes
- Inverse Document Frequency? : yes
- Other Term Weights? : yes
- Information Theoretic Weights? : yes
- Document Length? : yes

Machine Information

- Machine Type for TREC Experiment: Sun SPARC 10
- Was the Machine Dedicated or Shared: Shared
- Amount of Hard Disk Storage (in MB): 9000
- Amount of RAM (in MB): 80
- Clock Rate of CPU (in MHz): 55

System Summary and Timing
Organization Name: Institute of Systems Science
List of Run ID's: issah1 issah2

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list: 495
- Controlled Vocabulary? : NO
- Stemming Algorithm: Porter
  - Morphological Analysis: NO
- Term Weighting: YES
- Phrase Discovery? :
- Tokenizer? : YES
  - Patterns which are tokenized: WORDS
- Other Techniques for building Data Structures: YES, sentence masking

Statistics on Data Structures built from TREC Text

- Inverted index
  - Run ID : issah1 issah2
  - Total Storage (in MB): 1385
  - Total Computer Time to Build (in hours): 17.5
  - Automatic Process? (If not, number of manual hours): YES
  - Use of Term Positions? : NO
  - Only Single Terms Used? : YES
- Clusters
- N-grams, Suffix arrays, Signature Files
- Knowledge Bases
  - Use of Manual Labor
- Special Routing Structures
- Other Data Structures built from TREC text

Data Built from Sources Other than the Input Text

- Internally-built Auxiliary File
  - Use of Manual Labor
- Externally-built Auxiliary File

Query construction

Automatically Built Queries (Ad-Hoc)

- Topic Fields Used: title, description
- Average Computer Time to Build Query (in cpu seconds): 0.0867
- Method used in Query Construction
  - Term Weighting (weights based on terms in topics)? : YES
  - Tokenizer? :
  - Expansion of Queries using Previously-Constructed Data Structure? :

Automatically Built Queries (Routing)

- Method used in Query Construction
  - Terms Selected From
  - Term Weighting with Weights Based on terms in
  - Phrase Extraction from

  - Syntactic Parsing
  - Word Sense Disambiguation using
  - Proper Noun Identification Algorithm from
  - Tokenizer
  - Heuristic Associations to Add Terms from
  - Expansion of Queries using Previously-Constructed Data Structure:
  - Automatic Addition of Boolean connectors or Proximity Operators using information from

Manually Constructed Queries (Ad-Hoc)

- Topic Fields Used: title, description
- Average Time to Build Query (in Minutes): 0.5
- Type of Query Builder
  - Computer System Expert: YES
- Tools used to Build Query
  - Knowledge Base Browser? :
  - Other Lexical Tools? :
- Method used in Query Construction
  - Term Weighting? : YES
  - Addition of Terms not Included in Topic? :
  - Other: PRUNING, to reduce search times

Manually Constructed Queries (Routing)

- Type of Query Builder
- Tools used to Build Query
  - Knowledge Base Browser? :
  - Other Lexical Tools? :
- Data Used for Building Query from
- Method used in Query Construction
  - Addition of Terms not Included in Topic? :

Interactive Queries

- Type of Person doing Interaction
- Average Time to do Complete Interaction
- Methods used in Interaction
  - Automatic Query Expansion from Relevant Documents? :
  - Manual Methods

Searching

Search Times

  - Run ID : issah1
  - Computer Time to Search (Average per Query, in CPU seconds): 78.6
- Search Times
  - Run ID : issah2
  - Computer Time to Search (Average per Query, in CPU seconds): 32.4

Machine Searching Methods

- Machine Searching Methods
  - Probabilistic Model? : YES

Factors in Ranking

- Factors in Ranking
  - Term Frequency? : YES
  - Inverse Document Frequency? : YES
  - Other Term Weights? : YES, based on query field
  - Proximity of Terms? : YES
  - Document Length? : Y

Machine Information

- Machine Type for TREC Experiment: Sun SPARC20 workstation
- Was the Machine Dedicated or Shared: shared
- Amount of Hard Disk Storage (in MB): 10,000
- Amount of RAM (in MB): 96
- Clock Rate of CPU (in MHz): 50

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: 13 man MONTHS
- Given appropriate resources
  - Could your system run faster? : NO
- Features the System is Missing that would be beneficial: sophisticated tokenizers, thesaurus expansion

Significant Areas of System

- Brief Description of features in your system which you feel impact the system and are not answered by above questions: full multilingual architecture, maximum memory usage of 2MB per query, updates to corpus reflected within 5 mins

System Summary and Timing
Organization Name: APLab, SCILS, Rutgers
List of Run ID's: rutscn20

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list: no
- Controlled Vocabulary? : no
- Stemming Algorithm:
  - Morphological Analysis: no
- Term Weighting: no
- Phrase Discovery? :
  - Kind of Phrase: no
  - Method Used (statistical, syntactic, other): no
- Syntactic Parsing? : no
- Word Sense Disambiguation? : no
- Heuristic Associations (including short definition)? : no
- Spelling Checking (with manual correction)? : no
- Spelling Correction? : no
- Proper Noun Identification Algorithm? : no
- Tokenizer? :
  - Patterns which are tokenized: no
- Manually-Indexed Terms? : no
- Other Techniques for building Data Structures: 5-grams

Statistics on Data Structures built from TREC Text

- Inverted index
  - Run ID : no
- Clusters
  - Run ID : no
- N-grams, Suffix arrays, Signature Files
  - Run ID : rutscn20
  - Total Storage (in MB): no
  - Total Computer Time to Build (in hours): no database built
  - Automatic Process? (If not, number of manual hours): n/a
  - Brief Description of Method: scanning
- Knowledge Bases
  - Run ID : no
  - Use of Manual Labor
- Special Routing Structures
  - Run ID : no
- Other Data Structures built from TREC text
  - Run ID : no

Data Built from Sources Other than the Input Text

- Internally-built Auxiliary File
  - Domain (independent or specific): no
  - Use of Manual Labor
- Externally-built Auxiliary File
  - Type of File (Treebank, WordNet, etc.): no

Query construction

Automatically Built Queries (Ad-Hoc)

- Topic Fields Used: no
- Method used in Query Construction
  - Tokenizer? :
  - Expansion of Queries using Previously-Constructed Data Structure? :

Automatically Built Queries (Routing)

- Topic Fields Used: no
- Method used in Query Construction
  - Terms Selected From
  - Term Weighting with Weights Based on terms in
  - Phrase Extraction from
  - Syntactic Parsing
  - Word Sense Disambiguation using
  - Proper Noun Identification Algorithm from
  - Tokenizer
  - Heuristic Associations to Add Terms from
  - Expansion of Queries using Previously-Constructed Data Structure:
  - Automatic Addition of Boolean connectors or Proximity Operators using information from

Manually Constructed Queries (Ad-Hoc)

- Topic Fields Used: all
- Average Time to Build Query (in Minutes): 2
- Type of Query Builder
- Tools used to Build Query
  - Knowledge Base Browser? :
  - Other Lexical Tools? :
- Method used in Query Construction
  - Addition of Terms not Included in Topic? :
  - Other: manual elimination of all stop words and non-content words

Searching

Search Times

  - Run ID : rutscn20
  - Computer Time to Search (Average per Query, in CPU seconds): 18,000 using nawk
  - Component Times : automated construction of scanning script 0%, scanning 100%

Machine Searching Methods

  - N-gram Matching? : yes
  - Free Text Scanning? : yes

Factors in Ranking

  - Term Frequency? : yes
  - Inverse Document Frequency? : no
  - Document Length? : yes
  - N-gram Frequency? : yes
  - Other: Partial match 5-grams, with any 4 of 5 characters correct.

Machine Information

- Machine Type for TREC Experiment: Sun SparcStation 20
- Was the Machine Dedicated or Shared: mostly dedicated
- Amount of Hard Disk Storage (in MB): 9,000
- Amount of RAM (in MB): 48
- Clock Rate of CPU (in MHz): 50

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: a few hours
- Given appropriate resources
  - Could your system run faster? : yes
  - By how much (estimate)? : 100 times probably
- Features the System is Missing that would be beneficial: The list is endless.

Significant Areas of System

- Brief Description of features in your system which you feel impact the system and are not answered by above questions: Partial match 5-grams to provide some level of robustness against data corruption.

System Summary and Timing
Organization Name: Department of Computing Science, University of Glasgow
List of Run ID's: GLAIR1

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list: 320
- Controlled Vocabulary? : No
- Stemming Algorithm: Porter
  - Morphological Analysis: None
- Term Weighting: tf*idf
- Phrase Discovery? :
- Tokenizer? : Simple word boundary tokeniser

Statistics on Data Structures built from TREC Text

- Inverted index
  - Run ID : GLAIR1
  - Total Storage (in MB): ~100
  - Total Computer Time to Build (in hours): 30
  - Automatic Process? (If not, number of manual hours): Yes
  - Use of Term Positions? : No
  - Only Single Terms Used? : Yes
- Clusters
- N-grams, Suffix arrays, Signature Files
- Knowledge Bases
  - Use of Manual Labor
- Special Routing Structures
- Other Data Structures built from TREC text

Query construction

Automatically Built Queries (Ad-Hoc)

- Topic Fields Used: description
- Average Computer Time to Build Query (in cpu seconds): very short time
- Method used in Query Construction
  - Tokenizer? Simple word boundary tokeniser:
  - Expansion of Queries using Previously-Constructed Data Structure? :

Searching

Search Times

- Run ID : GLAIR1
- Computer Time to Search (Average per Query, in CPU seconds): <1 minute?

Machine Searching Methods

- Probabilistic Model? : Yes

Factors in Ranking

- Term Frequency? : Yes
- Inverse Document Frequency? : Yes
- Document Length? : Yes

Machine Information

- Machine Type for TREC Experiment: SPARC 10
- Was the Machine Dedicated or Shared: Shared
- Amount of Hard Disk Storage (in MB): 9Gb
- Amount of RAM (in MB): 32Mb
- Clock Rate of CPU (in MHz): Don't know

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: 2 furlongs per fortnight

System Summary and Timing
Organization Name: HNC Software Inc.
List of Run ID's: HNC11, HNC21

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list: 375
- Stemming Algorithm: Lovins
  - Morphological Analysis: No
- Term Weighting: Yes
- Phrase Discovery? :
  - Kind of Phrase: Yes
  - Method Used (statistical, syntactic, other): statistical
- Syntactic Parsing? : No
- Word Sense Disambiguation? : No
- Heuristic Associations (including short definition)? : No
- Spelling Checking (with manual correction)? : No
- Spelling Correction? : No
- Proper Noun Identification Algorithm? : No
- Tokenizer? : Yes
  - Patterns which are tokenized: No
- Manually-Indexed Terms? : No

Statistics on Data Structures built from TREC Text

- Inverted index
- Clusters
- N-grams, Suffix arrays, Signature Files
- Knowledge Bases
  - Use of Manual Labor
- Other Data Structures built from TREC text
  - Run ID : HNC11
  - Type of Structure: Word Context Vectors
  - Total Storage (in MB): 300
  - Total Computer Time to Build (in hours): 130
  - Automatic Process? (If not, number of manual hours): Yes
- Other Data Structures built from TREC text
  - Run ID : HNC21
  - Type of Structure: Word Context Vectors
  - Total Storage (in MB): 300
  - Total Computer Time to Build (in hours): 130
  - Automatic Process? (If not, number of manual hours): Yes

Data Built from Sources Other than the Input Text

- Internally-built Auxiliary File
  - Use of Manual Labor
- Externally-built Auxiliary File

Query construction

Automatically Built Queries (Ad-Hoc)

- Method used in Query Construction
  - Tokenizer? :

    - Expansion of Queries using Previously-Constructed Data Structure? :

Automatically Built Queries (Routing)

- Average Computer Time to Build Query (in cpu seconds): 10-20
- Method used in Query Construction
    - Terms Selected From
        - Only Documents with Relevance Judgments: Yes
    - Term Weighting with Weights Based on terms in
    - Phrase Extraction from
    - Syntactic Parsing
    - Word Sense Disambiguation using
    - Proper Noun Identification Algorithm from
    - Tokenizer
    - Heuristic Associations to Add Terms from
    - Expansion of Queries using Previously-Constructed Data Structure:
    - Automatic Addition of Boolean connectors or Proximity Operators using information from

Manually Constructed Queries (Ad-Hoc)

- Type of Query Builder
- Tools used to Build Query
    - Knowledge Base Browser? :
    - Other Lexical Tools? :
- Method used in Query Construction
    - Addition of Terms not Included in Topic? :

Manually Constructed Queries (Routing)

- Type of Query Builder
- Tools used to Build Query
    - Knowledge Base Browser? :
    - Other Lexical Tools? :
- Data Used for Building Query from
- Method used in Query Construction
    - Addition of Terms not Included in Topic? :

Interactive Queries

- Type of Person doing Interaction
- Average Time to do Complete Interaction
- Methods used in Interaction
    - Automatic Query Expansion from Relevant Documents? :
    - Manual Methods

Searching

Search Times

- Run ID : HNC11
- Computer Time to Search (Average per Query, in CPU seconds): 20
- Component Times : Context Vector dot product sorting

Machine Searching Methods

- Machine Searching Methods
    - Vector Space Model? : Yes

Machine Information

- Machine Type for TREC Experiment: Sun Sparc 10
- Was the Machine Dedicated or Shared: Shared
- Amount of Hard Disk Storage (in MB): 2 GB
- Amount of RAM (in MB): 512
- Clock Rate of CPU (in MHz): 45

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: 3-4 years
- Given appropriate resources
  - Could your system run faster? : Yes
  - By how much (estimate)? : 2 or 3 times on faster hardware

Significant Areas of System

- Brief Description of features in your system which you feel impact the system and are not answered by above questions: Routing method uses an LVQ (neural network) algorithm given the Context Vectors of judged documents to create Query Context Vector(s)

System Summary and Timing
Organization Name: NEC
List of Run ID's: virtu3, virtu4

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list: 430
- Controlled Vocabulary? : Yes
- Stemming Algorithm: Yes
  - Morphological Analysis: Yes
- Term Weighting: Yes
- Phrase Discovery? : No
- Syntactic Parsing? : No
- Word Sense Disambiguation? : No
- Heuristic Associations (including short definition)? : No
- Spelling Checking (with manual correction)? : No
- Spelling Correction? : No
- Proper Noun Identification Algorithm? : No
- Tokenizer? : Yes
  - Patterns which are tokenized: common patterns
- Manually-Indexed Terms? : No
- Other Techniques for building Data Structures: No

Statistics on Data Structures built from TREC Text

- Inverted index
  - Run ID : virtu3, virtu4
  - Total Storage (in MB): 3000
  - Total Computer Time to Build (in hours): 200
  - Automatic Process? (If not, number of manual hours): Yes
  - Use of Term Positions? : Yes
  - Only Single Terms Used? : Yes
- Clusters
  - Run ID : No
- N-grams, Suffix arrays, Signature Files
  - Run ID : No
- Knowledge Bases
  - Run ID : No
  - Use of Manual Labor
- Special Routing Structures
  - Run ID : No
- Other Data Structures built from TREC text
  - Run ID : virtu3
  - Type of Structure: word co-occurrency
  - Total Storage (in MB): 630
  - Total Computer Time to Build (in hours): 120
  - Automatic Process? (If not, number of manual hours): Yes
  - Brief Description of Method: Frequency of two words occuring in the same paragraph.

Query construction

Automatically Built Queries (Ad-Hoc)

- Topic Fields Used: All
- Average Computer Time to Build Query (in cpu seconds): 20 min per query

- Method used in Query Construction
  - Term Weighting (weights based on terms in topics)? : Yes
  - Phrase Extraction from Topics? : Yes
  - Syntactic Parsing of Topics? : Yes
  - Word Sense Disambiguation? : No
  - Proper Noun Identification Algorithm? : Yes
  - Tokenizer? : Yes
    - Patterns which are Tokenized: part of noun phrase identification
  - Heuristic Associations to Add Terms? : No
  - Expansion of Queries using Previously-Constructed Data Structure? : Yes
    - Structure Used: thesaurus (WordNet)
  - Automatic Addition of Boolean Connectors or Proximity Operators? : No
  - Other: No

Automatically Built Queries (Routing)

- Topic Fields Used: All
- Average Computer Time to Build Query (in cpu seconds): 30 min per query
- Method used in Query Construction
  - Terms Selected From
    - Topics: All
    - All Training Documents: No
    - Only Documents with Relevance Judgments: No
  - Term Weighting with Weights Based on terms in
    - Topics: Yes
    - All Training Documents: No
    - Documents with Relevance Judgments: Yes
  - Phrase Extraction from
    - Topics: Yes
    - All Training Documents: No
    - Documents with Relevance Judgments: No
  - Syntactic Parsing
    - Topics: Yes
    - All Training Documents: No
    - Documents with Relevance Judgments: No
  - Word Sense Disambiguation using
    - Topics: No
    - All Training Documents: No
    - Documents with Relevance Judgments: No
  - Proper Noun Identification Algorithm from
    - Topics: Yes
    - All Training Documents: No
    - Documents with Relevance Judgments: No
  - Tokenizer
    - Patterns which are tokenized (dates, phone numbers, common patterns, etc): part of noun phrase identificaton
    - from Topics: Yes
    - from All Training Documents: No
    - from Documents with Relevance Judgments: No
  - Heuristic Associations to Add Terms from
    - Topics: No
    - All Training Documents: No
    - Documents with Relevance Judgments: No
  - Expansion of Queries using Previously-Constructed Data Structure:
    - Structure Used: word co-occurrency
  - Automatic Addition of Boolean connectors or Proximity Operators using information from
    - Topics: No
    - All Training Documents: No

- Documents with Relevance Judgments: No

Searching

Search Times

- Run ID : virtu3, virtu4
- Computer Time to Search (Average per Query, in CPU seconds): 1200

Machine Searching Methods

- Vector Space Model? : Yes
- Probabilistic Model? : No
- Cluster Searching? : No
- N-gram Matching? : No
- Boolean Matching? : No
- Fuzzy Logic? : No
- Free Text Scanning? : No
- Neural Networks? : No
- Conceptual Graph Matching? : No
- Other: No

Factors in Ranking

- Term Frequency? : Yes
- Inverse Document Frequency? : No
- Other Term Weights? : No
- Semantic Closeness? : No
- Position in Document? : No
- Syntactic Clues? : No
- Proximity of Terms? : No
- Information Theoretic Weights? : No
- Document Length? : Yes
- Percentage of Query Terms which match? : No
- N-gram Frequency? : No
- Word Specificity? : No
- Word Sense Frequency? : No
- Cluster Distance? : No
- Other: No

Machine Information

- Machine Type for TREC Experiment: sparc10
- Was the Machine Dedicated or Shared: shared
- Amount of Hard Disk Storage (in MB): 10000
- Amount of RAM (in MB): 128
- Clock Rate of CPU (in MHz): 40

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: Three people in two month
- Given appropriate resources
  - Could your system run faster? : Yes
  - By how much (estimate)? : 50%
- Features the System is Missing that would be beneficial: The combined use of thesaurus and word co-occurrence information

```
System Summary and Timing
Organization Name:  University of Kansas
List of Run ID's:  KU1

Construction of Indices, Knowledge Bases, and other Data Structures

  Methods Used to build Data Structures

  - Length (in words) of the stopword list: 23
  - Controlled Vocabulary? : no
  - Stemming Algorithm: no
    - Morphological Analysis: no
  - Term Weighting:  yes
  -  Phrase Discovery? :
  -  Heuristic Associations (including short definition)? : yes
  -  Tokenizer? :

  Statistics on Data Structures built from TREC Text

  - Inverted index
    - Run ID : KU1
    - Total Storage (in MB): 325
    - Total Computer Time to Build (in hours): 11 hours
    - Automatic Process? (If not, number of manual hours): yes
    - Use of Term Positions? : no
    - Only Single Terms Used? : yes
  - Clusters
  - N-grams, Suffix arrays, Signature Files
  - Knowledge Bases
    - Run ID : KU1
    - Total Storage (in MB): 196
    - Total Computer Time to Build (in hours):  43 hours
    - Automatic Process? (If not, number of manual hours): yes
    - Use of Manual Labor
    - Number of Concepts Represented: 10022
    - Type of Representation:  similarity matrix
    - Auxiliary Files Needed: none
  - Special Routing Structures
  - Other Data Structures built from TREC text

Query construction

  Automatically Built Queries (Ad-Hoc)

  - Topic Fields Used: wsj:  LP, TEXT; sjm: LEADPARA, TEXT
  - Average Computer Time to Build Query (in cpu seconds):  2.4 minutes
    elapsed time (cpu time unavailable)
  - Method used in Query Construction
    - Term Weighting (weights based on terms in topics)? : yes
    - Phrase Extraction from Topics? : no
    - Syntactic Parsing of Topics? :no
    - Word Sense Disambiguation? : no
    - Proper Noun Identification Algorithm? :no
    - Tokenizer? :  no
    - Heuristic Associations to Add Terms? :yes
```

- Expansion of Queries using Previously-Constructed Data Structure? : yes
  - Structure Used: similarity matrix
- Automatic Addition of Boolean Connectors or Proximity Operators? :no

Searching

Search Times

- Run ID : KU1
- Computer Time to Search (Average per Query, in CPU seconds): 144
- Component Times : query expansion 10 document retrieval 134

Factors in Ranking

- Factors in Ranking
  - Term Frequency? : yes
  - Inverse Document Frequency? : yes
  - Other Term Weights? : yes
  - Semantic Closeness? : no
  - Position in Document? : no
  - Syntactic Clues? : no
  - Proximity of Terms? : no
  - Information Theoretic Weights? : no
  - Document Length? : yes
  - Percentage of Query Terms which match? : yes
  - N-gram Frequency? : no
  - Word Specificity? : no
  - Word Sense Frequency? : no
  - Cluster Distance? : no
  - Other: Term similarity between original term and terms added from similarity matrix by automatic expansion

Machine Information

- Machine Type for TREC Experiment: Sun SPARC 10
- Was the Machine Dedicated or Shared: Shared
- Amount of Hard Disk Storage (in MB): 9 GB
- Amount of RAM (in MB): 128
- Clock Rate of CPU (in MHz): 50

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: modest
- Given appropriate resources
  - Could your system run faster? : yes
  - By how much (estimate)? : 20%
- Features the System is Missing that would be beneficial: disambiguation, browser for viewing term similarity matrix

Significant Areas of System

- Brief Description of features in your system which you feel impact the system and are not answered by above questions: automatic calculation of term similarity based on the contexts in the corpus in which the terms instances appear.

System Summary and Timing
Organization Name: University of Neuchatel (Switzerland)
List of Run ID's: UniNE3, UniNE4

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list: 571 (SMART)
- Stemming Algorithm: yes Lovins (SMART)
- Term Weighting: yes
- Phrase Discovery? :
- Tokenizer? :

Statistics on Data Structures built from TREC Text

- Inverted index
  - Run ID : UniNE3, UniNE4
  - Total Storage (in MB): 179 UniNE3, 710 UniNE4
  - Total Computer Time to Build (in hours): 2.6 UniNE3, 10 UniNE4
  - Automatic Process? (If not, number of manual hours): yes
  - Only Single Terms Used? : yes
- Clusters
- N-grams, Suffix arrays, Signature Files
- Knowledge Bases
  - Use of Manual Labor
- Special Routing Structures
- Other Data Structures built from TREC text

Query construction

Automatically Built Queries (Ad-Hoc)

- Topic Fields Used: desc
- Average Computer Time to Build Query (in cpu seconds): 0.3
- Method used in Query Construction
  - Term Weighting (weights based on terms in topics)? : yes
  - Tokenizer? :
  - Expansion of Queries using Previously-Constructed Data Structure? :

Searching

Search Times

- Run ID : UniNE3, UniNE4
- Computer Time to Search (Average per Query, in CPU seconds): 15.5

Machine Searching Methods

- Vector Space Model? : yes
- Probabilistic Model? : yes

Factors in Ranking

- Term Frequency? : yes
- Inverse Document Frequency? : yes

  - Other Term Weights? : normalization of seach term weights done by SMART
  - Document Length? : yes

Machine Information

- Machine Type for TREC Experiment: SUN SPARCstation 10 model 51
- Was the Machine Dedicated or Shared: shared
- Amount of Hard Disk Storage (in MB): 6,144 MB
- Amount of RAM (in MB): 128 MB
- Clock Rate of CPU (in MHz): 50 MHz

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: 4 months to understand SMART, 6 months to write our additional features in Smalltalk-80
- Given appropriate resources
  - Could your system run faster? : yes, because Smalltalk is interpreted
  - By how much (estimate)? : the improvement factor is unknown

System Summary and Timing
Organization Name: University of Virginia
List of Run ID's: drift1 drift2

Construction of Indices, Knowledge Bases, and other Data Structures

Methods Used to build Data Structures

- Length (in words) of the stopword list: 571
- Controlled Vocabulary? : no
- Stemming Algorithm: yes, triestem from SMART v11.0
- Term Weighting: tfc documents, nfx queries
- Phrase Discovery? :
- Tokenizer? :

Statistics on Data Structures built from TREC Text

- Inverted index
- Clusters
- N-grams, Suffix arrays, Signature Files
- Knowledge Bases
  - Use of Manual Labor
- Special Routing Structures
- Other Data Structures built from TREC text
  - Run ID : drift1 and drift2
  - Type of Structure: Document vectors
  - Total Storage (in MB): 190
  - Total Computer Time to Build (in hours): 7
  - Automatic Process? (If not, number of manual hours): yes
  - Brief Description of Method: TREC text is run through Smart to remove stop words and do word stemming. Smart vectors are then converted to Drift format.

Query construction

Automatically Built Queries (Routing)

- Topic Fields Used: all
- Average Computer Time to Build Query (in cpu seconds): less than 1
- Method used in Query Construction
  - Terms Selected From
    - Topics: yes
    - All Training Documents: yes
  - Term Weighting with Weights Based on terms in
    - All Training Documents: yes
  - Phrase Extraction from
  - Syntactic Parsing
  - Word Sense Disambiguation using
  - Proper Noun Identification Algorithm from
  - Tokenizer
  - Heuristic Associations to Add Terms from
  - Expansion of Queries using Previously-Constructed Data Structure:
  - Automatic Addition of Boolean connectors or Proximity Operators using information from

Searching

Search Times

- Run ID : drift2
- Computer Time to Search (Average per Query, in CPU seconds): 150
- Component Times : 2% query broadcast to sites 88% local search 10% result merge

Machine Searching Methods

- Vector Space Model? : yes

Factors in Ranking

- Term Frequency? : yes
- Inverse Document Frequency? : yes
- Document Length? : yes

Machine Information

- Machine Type for TREC Experiment: Sun SPARCserver 20
- Was the Machine Dedicated or Shared: shared CPU, dedicated disk
- Amount of Hard Disk Storage (in MB): 2000
- Amount of RAM (in MB): 128
- Clock Rate of CPU (in MHz): 50

System Comparisons

- Amount of "Software Engineering" which went into the Development of the System: System was built by 1 person, half-time in 6 months.
- Given appropriate resources
  - Could your system run faster? : Yes
  - By how much (estimate)? : Factor of 10-20
- Features the System is Missing that would be beneficial:
  Hardware: More disk space, dedicated CPU.
  Software: Inverted index, thesaurus, query expansion, you name it.

Significant Areas of System

- Brief Description of features in your system which you feel impact the system and are not answered by above questions: Drift simulates a distributed IR system. Search time, memory usage, and disk usage all increase as the number of sites in the system increases.

APPENDIX C

SUMMARY
PERFORMANCE COMPARISONS
TREC-2, TREC-3, TREC-4

Karen Sparck Jones, November 7, 1995

The performance comparisons between TREC-2 and TREC-3 appeared as Appendix C to the TREC-3 Proceedings, i.e. to 'Overview of the Third Text REtrieval Conference (TREC-3)' Ed D.K. Harman, NIST Special Publication 500-225, National Institute of Standards and Technology, 1995 (and see also the opening Overview paper by Donna Harman, p 18).

The TREC-4 figures are from the TREC-4 conference working papers.

The tables present Precision performance at Document Cutoff 30.

The data are only for full Category A runs, not Category B, and cover only the higher levels of performance, not all the submitted runs.

The conventions are as follows: figures are not rounded; performance is assigned to 'blocks'; teams per block are in Proceedings order; the best of two official runs is taken where there are two, regardless of the retrieval method used (thus queries may be automatically or manually formed). The comparisons are for ad hoc, non-interactive searching, for routing, and for interactive searching (first tested for TREC-3), respectively: other TREC-4 test tracks had relatively few players and/or are not directly comparable and so are omitted here.

DOCUMENTS 30

	ADHOC			ROUTING			INTERACTIVE ('Routing'-style TREC-3, Task 2 TREC-4)	
	TREC2	TREC3	TREC4	TREC2	TREC3	TREC4	TREC3	TREC4
>=60		UMass City Berkeley		Cornell Dortmund	City			
>=55	UMass HNC VT	Cornell Mead		City Berkeley UMass Bellcore CMU CUNY	UMass Cornell Berkeley Dortmund Bellcore	City UMass Xerox		
>=50	Cornell Berkeley Dortmund CMU Verity Siemens CUNY	VT Westlaw ETH CUNY		Rutgers HNC GE TRW Verity Siemens	CMU Westlaw Logicon TRW Florida	Cornell CUNY		Cornell
>=45	City Bellcore ETH CITRI Conquest	NYU CMU RMIT RutgersK	Excalibur CUNY Waterloo	VT	Xerox NYU Verity ETH NSA NEC	Logicon GE/NYU	RutgersB Verity	Rutgers City
>=40			Berkeley Clarit/CMU Cornell GMU UMass InText ANU				City	
>=35			City GE/NYU					

NOTES:

a) UMass was formerly labelled Amherst.
b) RMIT was joined with CITRI in TREC-2.
c) RutgersK was one of two Rutgers groups in TREC-3.
d) Excalibur merged with Conquest during TREC-4.
e) While some groups recur in the tables, others have varied in participation in TREC or have not always done well enough (whether through goofs or deliberate (but unprofitable) experiment) to figure in the upper performance levels selected for the tables. Some of the TREC-4 participants also concentrated on tracks not included here.

COMMENTS:

1) Ad hoc best performance improved from TREC-2 to TREC-3, even though the TREC-3 topics (source requests) were less rich. The sharp fall in performance for TREC-4 must reflect the very minimal requests supplied for the tests.

2) Routing performance also fell slightly for TREC-4, apparently reflecting a tough set of test topics (though these were still rich). The smaller number of groups must be attributed to participants' diversion to other test tracks (compared with ad hoc which, though not compulsory for all players, was strongly recommended).

3) While upper-level ad hoc performance in TREC-3 was as good as routing, showing that rich starting topics can be almost as useful as extensive training data, the difference between the two in TREC-4 is, not surprisingly, quite marked.

4) In general, constant participants who initially performed well have continued to do so, though some others who started less well have successfully improved their performance.

5) Interactive performance remains disconcertingly poor.

OVERALL REMARKS:

1) The General Findings for TREC-2 given in my 'Reflections on TREC' (Information Processing and Management 31, 1995, 291-314), and noted as continuing to apply in TREC-3 in my previous comparison tables, remain true for TREC-4.

2) Many (very) different approaches give similar performance.

3) Good performance is obtained with variants of the statistical approach souped up with phrases etc: there indeed seems to be an emerging consensus on the right weighting concepts, and on the value of phrases (even statistically defined ones).

4) Routing performance seems to have reached a plateau, in that provided there is a good supply of training data and fairly well-developed starting queries, a very respectable performance level can be achieved.
NOTE ESPECIALLY, ALL of the TREC-4 routing results above were obtained with automatically developed queries.

5) The TREC-4 ad hoc performance, on the other hand, is clearly more representative of normal retrieval situations than the earlier results. MORE SPECIFICALLY, there was a clear shift between TREC-3 and TREC-4 from automatic to manual query as the normal route to query development, doubtless reflecting a perceived need to beef up the search input. However Cornell, the only high performer with automatic query in TREC-3, remained in the top groups for TREC-4.
NOTE ALSO that the definition 'manual' covers a wide range of human effort from the fairly minimal to the very intensive, though it seems that relatively modest effort can deliver as well as much more intensive work.

6) There are too few relevant interactive results for TREC-4 (i.e. ones comparable with those for TREC-3) to draw any useful conclusions about why interactive searching does not upgrade performance in a more striking way.

7) OVERALL, the difference between the best performance blocks for ad hoc >= 45 % P ; routing >= 55 ; interactive >= 50 % is not very large (equivalent to 13.5, 16.5 or 15 relevant documents respectively), and best automatic performance for ad hoc delivers >= 40 against manual >= 45 (equivalent to 12 as opposed to 13.5 relevant documents).

AS BEFORE, I EMPHASISE that

8) ALL of these points are broad brush ones: there may be significant differences

within performance blocks, and also none between members of adjoining blocks. However even without accepting the degree of similarity between runs that Jean Tague's TREC-3 Scheffe tests show, obliterating many of my block separations, and while also recognising that a Precision difference between 45 and 60 at Document cutoff may be not only statistically significant but meaningful to a user, I think a conservative view of performance differences is in order. Thus the really important point is (2) above, which applies both to performance in general and to best performance as exhibited in the tables.

*U.S. GOVERNMENT PRINTING OFFICE: 1996-417-392/67909

ANNOUNCEMENT OF NEW PUBLICATIONS ON COMPUTER SYSTEMS TECHNOLOGY

Superintendent of Documents
Government Printing Office
Washington, DC 20402

Dear Sir:

Please add my name to the announcement list of new publications to be issued in the series: National Institute of Standards and Technology Special Publication 500–.

Name ____

Company ____

Address ____

City ____________________ State ________________ Zip Code ______________

(Notification key N-503)

NIST Technical Publications

Periodical

Journal of Research of the National Institute of Standards and Technology—Reports NIST research and development in those disciplines of the physical and engineering sciences in which the Institute is active. These include physics, chemistry, engineering, mathematics, and computer sciences. Papers cover a broad range of subjects, with major emphasis on measurement methodology and the basic technology underlying standardization. Also included from time to time are survey articles on topics closely related to the Institute's technical and scientific programs. Issued six times a year.

Nonperiodicals

Monographs—Major contributions to the technical literature on various subjects related to the Institute's scientific and technical activities.

Handbooks—Recommended codes of engineering and industrial practice (including safety codes) developed in cooperation with interested industries, professional organizations, and regulatory bodies.

Special Publications—Include proceedings of conferences sponsored by NIST, NIST annual reports, and other special publications appropriate to this grouping such as wall charts, pocket cards, and bibliographies.

National Standard Reference Data Series—Provides quantitative data on the physical and chemical properties of materials, compiled from the world's literature and critically evaluated. Developed under a worldwide program coordinated by NIST under the authority of the National Standard Data Act (Public Law 90-396). NOTE: The Journal of Physical and Chemical Reference Data (JPCRD) is published bimonthly for NIST by the American Chemical Society (ACS) and the American Institute of Physics (AIP). Subscriptions, reprints, and supplements are available from ACS, 1155 Sixteenth St., NW, Washington, DC 20056.

Building Science Series—Disseminates technical information developed at the Institute on building materials, components, systems, and whole structures. The series presents research results, test methods, and performance criteria related to the structural and environmental functions and the durability and safety characteristics of building elements and systems.

Technical Notes—Studies or reports which are complete in themselves but restrictive in their treatment of a subject. Analogous to monographs but not so comprehensive in scope or definitive in treatment of the subject area. Often serve as a vehicle for final reports of work performed at NIST under the sponsorship of other government agencies.

Voluntary Product Standards—Developed under procedures published by the Department of Commerce in Part 10, Title 15, of the Code of Federal Regulations. The standards establish nationally recognized requirements for products, and provide all concerned interests with a basis for common understanding of the characteristics of the products. NIST administers this program in support of the efforts of private-sector standardizing organizations.

Order the **following** *NIST publications—FIPS and NISTIRs—from the National Technical Information Service, Springfield, VA 22161.*

Federal Information Processing Standards Publications (FIPS PUB)—Publications in this series collectively constitute the Federal Information Processing Standards Register. The Register serves as the official source of information in the Federal Government regarding standards issued by NIST pursuant to the Federal Property and Administrative Services Act of 1949 as amended, Public Law 89-306 (79 Stat. 1127), and as implemented by Executive Order 11717 (38 FR 12315, dated May 11, 1973) and Part 6 of Title 15 CFR (Code of Federal Regulations).

NIST Interagency Reports (NISTIR)—A special series of interim or final reports on work performed by NIST for outside sponsors (both government and nongovernment). In general, initial distribution is handled by the sponsor; public distribution is by the National Technical Information Service, Springfield, VA 22161, in paper copy or microfiche form.

U.S. Department of Commerce
National Institute of Standards
and Technology
Gaithersburg, MD 20899–0001

Official Business
Penalty for Private Use $300