

INFORMATION RETRIEVAL USING STATISTICAL CLASSIFICATION

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF STATISTICS
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

By
David A. Hull
November 1994

© Copyright 1994 by David A. Hull
All Rights Reserved

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Jerome Friedman
(Principal Adviser)

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Arthur Owen

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Jan Pedersen
Xerox Palo Alto Research Center

Approved for the University Committee on Graduate Studies:

Dean of Graduate Studies

Acknowledgements

Thanks to Jan for patiently sticking with me and keeping me directed towards the final goal, the completion of this work.

Thanks to Jerry for always being willing to help, no matter when I appeared at his door.

Thanks to Hinrich, Marti, and all my other colleagues in ISTL/QCA at Xerox PARC who have provided lots of inspiration and advice over the years.

Thanks to Natalie for all her love and support.

Abstract

In the classical information retrieval (IR) problem, the system must find all documents in a collection that are related to a topic defined by a user's query. A common approach to the IR problem is to represent documents and the query as vectors of term frequencies and rank the documents in the collection according to their inner product similarity with respect to the query. When a sample of evaluated documents is available in addition to the query (often called routing), the problem can be attacked using techniques based on statistical classification. In order for statistical classification to be a feasible approach, the system must produce a relatively small set of high quality feature variables. It turns out that individual words, due to their quantity and ambiguity, are not optimal features. Previous work has focused on a technique known as Latent Semantic Indexing (LSI), which applies the singular value decomposition to a term-document matrix and represents terms and documents by linear combinations of orthogonal indexing variables.

The research presented in this thesis accomplishes the following goals. It provides a thorough discussion of evaluation in information retrieval experiments. It introduces the concept of a local LSI decomposition. LSI is used separately on a set of documents in the local region surrounding each query, creating query-specific feature variables and making the LSI technique feasible for very large document collections. It applies the classification technique known as Discriminant Analysis to the routing problem and presents experimental results on two text collections. It demonstrates that using a local LSI decomposition improves retrieval performance and represents documents using a relatively small number of feature variables. It finds that Discriminant Analysis sometimes leads to additional performance gains but that more research is needed to determine the optimal size and shape of the local region.

Contents

Acknowledgements	iv
Abstract	v
1 An Introduction to Information Retrieval	1
1.1 The Information Access Spectrum	2
1.2 Important Definitions in IR	4
1.3 Forms of the Classical IR Problem	5
1.4 Nature of IR Data	7
1.5 IR as a Statistical Problem	8
1.6 Goals of this Thesis	9
1.7 Outline of Subsequent Chapters	10
2 The Basics of the IR System	13
2.1 The Inverted Index	14
2.2 Lexical Analysis	15
2.2.1 The Stop List	16
2.2.2 Stemming Algorithms	17
2.3 The Text Database System	18
3 Search Strategies in Information Retrieval	20
3.1 Boolean Retrieval	21
3.1.1 Drawbacks of Boolean Model	21
3.1.2 Improving the Boolean Model	23

3.2	The Vector Space Model	25
3.2.1	Using the VSM for retrieval	25
3.2.2	Term Weighting	27
3.2.3	Subtopics and Passage Retrieval	29
3.3	Relevance Feedback	30
3.3.1	Feedback using the VSM	31
3.3.2	Improving Feedback Performance	31
3.3.3	Assumptions of Relevance Feedback	32
3.3.4	Feedback using a Probabilistic Model	33
3.4	The Routing Problem	35
4	Feature Selection and Dimension Reduction	37
4.1	Problems with the VSM	37
4.2	Improving the VSM	39
4.2.1	Query Expansion	39
4.2.2	Word Sense Disambiguation	41
4.2.3	Phrase Modelling	42
4.3	Latent Semantic Indexing	43
4.3.1	Description of LSI	44
4.3.2	Benefits of Using LSI	46
4.3.3	Drawbacks of Using LSI	46
4.3.4	Using LSI for Information Retrieval	47
4.3.5	Performance of LSI in Retrieval Experiments	49
5	Evaluation	50
5.1	Test Collections	51
5.1.1	Important Characteristics of the Test Collection	51
5.1.2	Problems with Test Collections	53
5.1.3	The Alternative - User Studies	55
5.2	Performance Measures	57
5.2.1	Precision and Recall	57
5.2.2	Averaging Methods	59

5.3	Assumptions in Evaluation	59
5.3.1	Normalization	60
5.3.2	Problems with the DCV	61
5.3.3	Relevance Feedback and the P-R curve	62
5.4	Univariate Measures of Performance	64
5.4.1	P-R based	64
5.4.2	Other Measures	65
5.5	An Alternative Method of Evaluation	67
5.6	Hypothesis Testing	69
5.6.1	A Brief Introduction	70
5.6.2	Comparisons between two methods	71
5.6.3	The Significance Tests	71
5.6.4	Optimal two-sample tests in IR	73
5.6.5	Comparisons between more than two methods	74
5.6.6	General Comments	78
6	Classification Methods	80
6.1	Properties of the IR Problem	81
6.2	Discriminant Analysis	82
6.2.1	The Discriminant Rules	83
6.2.2	Linear vs. Quadratic	84
6.2.3	Regularized Discriminant Analysis	85
6.3	Logistic Regression	87
6.4	Optimal Separating Hyperplanes	89
6.5	Classification Trees	90
7	Retrieval Experiments: Cranfield	93
7.1	The Cranfield Collection	94
7.2	Baseline Experiments using the VSM	97
7.2.1	Implementation of Vector Space Search	98
7.2.2	Term Weighting and Basic Search	99
7.2.3	Baseline: Relevance Feedback	103

7.3	Baseline: Routing	105
7.4	LSI Experiments	106
7.4.1	Description of Method	107
7.4.2	Experimental Results	109
7.4.3	Routing: Detailed look at Evaluation	111
7.5	Routing and Classification	111
7.5.1	Initial Experiment	114
7.5.2	A Local LSI Representation	115
7.5.3	Experimental Results	117
7.6	Discussion	118
8	Retrieval Experiments: Wall Street Journal	121
8.1	The Wall Street Journal	122
8.2	Initial Experiments	124
8.2.1	Baseline from the VSM	124
8.2.2	LSI on the WSJ	125
8.3	Query-Specific Screening of the Collection	127
8.3.1	The Screening Process	130
8.3.2	Local LSI	133
8.4	Classification Experiments	134
8.4.1	LDA and Logistic Regression	135
8.4.2	Regularized Discriminant Analysis	137
8.5	Discussion	141
9	Conclusion	143
9.1	The Vector Space Model	144
9.2	Dimension Reduction and Classification	145
9.3	Evaluation and Retrieval Experiments	146
9.4	Extensions of this work	147
	Bibliography	150

Chapter 1

An Introduction to Information Retrieval

Information comes in many forms: news, financial data, scientific research, etc . . . and represents one of the most important commodities in the modern world. All major political, economic, and intellectual endeavors rely on the quick and easy flow of information through a variety of media, such as books, radio, television, and electronic networks. Modern computing and networking technology make it possible to organize, store, and transport large bodies of data with minimal effort anywhere in the world. Without question, we have moved into the information age.

With so much material so easily accessible, many organizations and individuals have realized that the real issue is no longer getting enough information, but sorting out what is useful to them from vast quantities of useless material. Information retrieval systems, computers and software designed to index, store, and provide easy access to data, are rapidly developing to meet this need. As one of their most important features, the systems provide search tools, algorithms which map an expression of the user's information need into a mathematical form which is then used to identify relevant material in the database.

Developing accurate search methods requires solving several challenging problems and will represent the core of the work in this thesis. While information can be stored and transmitted in many forms, including speech, images, and video, it is

written text that still predominates in most retrieval systems. In this work, we will focus on developing new and better search methods for text-based systems.

1.1 The Information Access Spectrum

People approach document collections with a broad range of different information needs. There is an entire spectrum of goals for information access ranging from general to specific requests. Some examples of potential requests to a retrieval system, moving from specific to general are:

- 1) Find the answer to this specific question.
- 2) Find the document with title A or all documents by author B.
- 3) Find all articles in the collection that refer to this topic.
- 4) Produce a summary of the contents of this collection on these topics.
- 5) Produce a summary of the contents of this collection.

All of these requests could be valid queries for a retrieval system, although most fall outside the domain of classical information retrieval. A traditional definition for information retrieval (IR) is [46]: “An information retrieval system informs [the user of] . . . the existence and whereabouts of documents relating to his request. [It] does not inform the user on the subject of his request.” Therefore, the information retrieval task is best defined by question (3), and the remainder of this thesis will be devoted to finding a reasonable solution to this type of request. In the years since this definition was presented, however, researchers have broadened the field of IR to address all of the questions given above and have applied a broad variety of statistical tools to the process. While the work presented here will concentrate on the traditional IR problem, approaches to the other questions will be briefly described below.

While question-answering systems, useful for (1) above, do not fall under the classical domain of IR, there has been some work on such systems in the field. Kupiec [45] has developed a system for answering closed-class questions by doing a syntactic

analysis of the query to identify noun phrases and then using pattern matching combined with other IR techniques to find documents that may contain the answer. The document collection used for this experiment is an on-line encyclopedia. Some of the linguistic tools applied in this system, such as part-of-speech tagging, boolean search with proximity constraints, and syntactic analysis, are often used in more general information retrieval systems.

The task of database retrieval, described by question (2) above, requires a somewhat different approach. Database queries are assumed to be precise, and the task of finding the relevant documents boils down to looking up the records which share the attributes defined in the query. Much of the work on database retrieval in IR is concerned with performance issues, since the retrieval task itself is well defined and the problem comes in scaling it to the size of full-text collections. In contrast, classical information retrieval (3) is a much more approximate science, since no sense of exact match exists between documents and queries in the IR setting. The value of a document is measured by its relevance to the query, which is often expressed in the framework of a probabilistic model. Most retrieval strategies are designed to provide approximate solutions and thus are not suitable for the database retrieval task. Work has been done by Fuhr [27] on combining IR with database systems.

One approach to the task defined by question (4) is known as text categorization. Assigning labels to documents from a group of predefined categories is a useful method for organizing a large text collection. Searchers can use these labels or key words to quickly focus their attention on the documents that relate to their questions. Traditionally, labels have been produced manually, representing a huge investment in time and effort. Hence, a system that automatically induces accurate category labels would save a great deal of human labor. The techniques used for solving this problem tend to come from machine learning and statistical classification, and many of them will be helpful for solving the traditional retrieval problem. If one thinks of each query as generating a distinct category topic, the information retrieval problem is strongly related to text categorization. Several recent papers have presented approaches to this problem. Lewis and Ringuette [49] examine models based on Bayesian estimation of word parameters and decision trees. Apte et. al. [4] use rule induction methods to

estimate category membership labels.

Cutting et. al. [18] describe the summarization task defined by question (5) as browsing. One can use the analogy of a textbook. The traditional IR task corresponds to looking up an item in the index, while browsing is more like scanning the table of contents. The browser is not so much interested in finding something specific as in learning about the contents of the collection. Of course a table of contents does not exist for the typical document collection, so it must be constructed either manually or automatically by the system. Therefore, the system requires a special set of tools to produce an effective summarization of large numbers of documents. In this context document summarization and presentation become important issues, as the user must be able to easily understand the relationship between a large number of documents. Cutting et. al. present a useful approach to this problem. Documents are grouped using cluster analysis, and a description of the contents of each document cluster is presented to the user. The user then has the option to select a subset of the clusters for further evaluation, and a new representation is built on this subset of the collection.

1.2 Important Definitions in IR

Before proceeding further, it is important to define several significant terms that will occur repeatedly in our discussion of the theory and implementation of IR systems. A *query* is a text expression that describes the information need of the user. The query is passed to the *retrieval system*, which uses this information to determine which documents in its collection are *relevant* to the user's request. A *document* is the organizational unit of the information collection. The *collection* consists of a large number of documents. A *relevant document* is one which contains information related to the query.

Document collections can take on a number of different forms. Some examples are given below:

	Collection	Document
(1)	Library	books, magazines, etc . . .
(2)	Journal	articles
(3)	Encyclopedia	entries
(4)	Newsgroup	postings
(5)	Newswire	news stories

A document collection can be static, such as an encyclopedia, or it can be dynamic, such as a newsgroup, with messages/documents appearing and disappearing over time.

A natural language query can range from a simple phrase such as “information superhighway” to a complete sentence such as: “Please find all references to cluster analysis in the context of information retrieval.” to much longer passages that may consist of several paragraphs or even several pages of text. It is a common strategy in information retrieval to incorporate the text of full documents into the query [69].

An IR system matches the query against the collection and returns a set of documents to the user, often ranked in order of their estimated relevance. The number of relevant documents retrieved provides an objective measure of system performance. This measure is usually normalized in one of two different fashions. *Precision* is defined as the number of relevant documents retrieved over the total number retrieved and *recall* is defined as the number of relevant documents retrieved over the total number relevant. Evaluation of system performance will be discussed in great detail in chapter 5.

1.3 Forms of the Classical IR Problem

The task of the information retrieval system is to match the query against the document collection and return relevant documents to the user. The success of the system will depend significantly on the quality and quantity of the information associated with the query. If a lot of data is available about what defines a relevant document, then the system may be able to employ more advanced categorization techniques

to the collection. For this reason, we divide the query-based IR problem into two categories, *ad hoc* search and routing [34].

In the *ad hoc* search problem, the system must rely solely on the initial query provided by the user. The system processes and matches the query against a known static collection of documents and returns a subset of these documents to the user. The problem is familiar from a number of settings. For example, a library user queries an on-line catalog system to find references about a topic of interest.

The routing problem assumes that more information is available to the system. In particular, there exists a set of documents that have already been classified as relevant or not relevant which the system can use to supplement the initial query. The system will then attempt to categorize unexamined documents to find a subset which is likely to be relevant. The routing problem often occurs in the processing of dynamic document collections. The user is assumed to have one or more standing queries which define continuing areas of interest and each new document that enters the system is matched against these queries. A simple example is a filter for a news reader. The user might provide a short description of the topic of interest as well as a sample of messages from the group which are about that topic. The system examines incoming messages and directs the relevant ones to the user's active folder.

The optimal response to a query would be for the system to find all the relevant documents and return nothing that is not relevant. This will almost never happen in practice, as the retrieval task can rarely be executed with such accuracy. However, the more documents that are returned to the user, the larger the number of relevant documents that will be found. But as the system considers documents that are less and less likely to be relevant, the percentage of non-relevant documents found will begin to increase. Therefore, it is clear that there is a trade-off between finding more relevant documents (recall) and being forced to examine fewer non-relevant documents (precision). The searcher must decide how many non-relevant documents he or she is willing to examine in order to discover an additional relevant document.

1.4 Nature of IR Data

For a retrieval system to operate effectively, there must be an algorithm that assesses whether a document is relevant to a query. For a human reader, a grasp of the relevant concept is a natural consequence of his or her inherent understanding of language. The reader follows the underlying meaning conveyed by the words in a document and can use that understanding to determine whether the document provides useful information about the query. For a computer, this is a far more difficult task. There is an ongoing debate in the artificial intelligence community as to whether computers will ever be capable of truly understanding human language.

There has been a great deal of work in the linguistic community on automatic language processing. This work has centered around the development of artificial structures that encode the relationship between words in the text. The goal of much of this work is to develop a system that by syntactic processing of the query can come to a basic understanding of the semantic content of the query. Unfortunately, such an understanding can rarely be reached from the text alone. Generally, the system must understand the unwritten context of the request, which requires substantial external knowledge of the domain being studied and may necessitate that a considerable amount of information from other domains be available as well [67].

Needless to say, it is extremely difficult to incorporate this knowledge into an information retrieval system. Even if the knowledge is available, evaluating even a simple query becomes a very time-consuming task. Therefore, natural language processing systems tend to be limited to extremely restricted domains. Most modern retrieval systems that are applied to text collections with a broad subject matter do not rely heavily on linguistic methods.

If the system cannot understand the text that it is processing, how can it determine if a query is related to a document? Part of the answer is to view the information retrieval problem in the context of database management. Database systems store and retrieve records using a relatively small number of fixed attributes. Each record is assigned values for some or all of these attributes. For example, a hospital might store all of its records using a small number of attributes, such as patient name,

social security number, illness or condition, attending doctor, etc . . . If one or more of these attributes is selected, the database system will return all records with the specified value of the attribute.

A basic information retrieval system can be designed in a similar fashion using words as attributes. Given a query, the IR system could retrieve all documents which have terms in common with the query. This is probably an inadequate model for long queries and it ignores higher-level linguistic structure, but it might work quite well for short one or two term requests. The database description of IR assumes that a document and a query are similar if they have some attribute (in this case one or more words) in common. We can generalize this prescription by claiming that the more terms the document and query have in common, the more likely it is that the document is related to the query. This leads to a mathematical model of document and term similarity and allows us to consider document retrieval as a statistical problem.

1.5 IR as a Statistical Problem

We can describe the general statistical classification problem as follows. Assume that there exists a population which can be divided into two or more groups. Each element of the population is described by a number of features. For some of the elements, their group membership is known; for some, it is unknown. The goal is to produce a classification rule which will accurately predict group membership information for those elements for which it is unknown. The classification rule will be generated by using the structure in the features.

It is easy to see how this formulation relates to the standard information retrieval problems. Documents are the elements of the population. For any particular query, there are two category labels, relevant and not relevant. Word frequencies can be used as features to characterise each document. We then wish to predict which documents are relevant as accurately as possible. While information retrieval fits quite easily into the framework of the classification problem, it has a number of unique features which make it a particularly challenging problem. They are outlined below.

While it seems like a natural and obvious choice to use terms as descriptive feature for a classification algorithm, this has a number of major drawbacks. The typical document collection has thousands to hundreds of thousands of unique words associated with it. It is extremely difficult to use all of this information to produce a general model of the relevant document. Furthermore, most terms occur in only a small fraction of the documents in the collection, so the utility of individual terms is limited. There is a vast amount of redundancy in language, which means that groups of terms or phrases exist which have basically the same meaning. There is also a lot of ambiguity in language, since the same word can often take on many different meanings depending on its context. All of these factors combine to limit the usefulness of terms in a classification algorithm.

The typical document collection contains a very large number of documents, often on a wide variety of subjects, so the scale of the problem alone is daunting. In addition, only a small fraction of documents will be relevant to any particular query, so it is often a real challenge to find a significant sample of relevant documents. The class of non-relevant documents is not really a class at all, since it consists of everything that is not related to the query, which spans all the other topics in the collection.

The system should be able to rank the documents in order of relevance, not merely make a binary decision between relevant and non-relevant. As mentioned before, the user must choose between fewer documents with a higher precision and more documents with a lower precision. In general, the system cannot balance the trade-offs without input from the user, because the number of documents that the user is willing to examine will generally depend on external factors not available to the system. Few readers will wish to examine a thousand or more documents, even if the system is convinced that they are all extremely relevant. All of these difficulties combine to make information retrieval an extremely challenging problem.

1.6 Goals of this Thesis

Among the difficulties associated with developing an improved information retrieval system is proving that it is measurably better than the current standard. The issue of

performance evaluation is a thorny topic in IR, and it probably has no unique solution. The first goal of this thesis will be to describe the current evaluation methodology and attempt to gain a better understanding of the underlying assumptions that support the standard performance metrics. We will also attempt to provide a more complete description of the different options for testing the hypothesis that one retrieval strategy is more effective than another.

As discussed in the previous section, a representation of documents as a set of term frequencies has a number of problems associated with it. Most effective retrieval systems currently in use rely on extremely simple term-matching strategies to identify relevant documents. To apply more sophisticated statistical models, we will have to find some practical method of dimensionality reduction. A retrieval strategy that must sift through hundreds of thousands of terms can only be of limited computational complexity if it is going to produce an answer in real time. In this thesis, we will evaluate the effectiveness of several methods of dimensionality reduction and suggest ways of using them even on the largest collections.

Once an adequate representation of documents has been found, one can consider adopting more sophisticated statistical classification algorithms. The final goal of this thesis will be to take several common classification techniques from statistics and machine learning and adapt them to the information retrieval problem. To make these methods work, we must overcome the fact that there will be an extremely limited number of relevant documents available to train the classification algorithms and that even the reduced document representations will require 100 or 200 dimensions.

1.7 Outline of Subsequent Chapters

The second chapter describes the basic components of the information retrieval system that convert raw text into the form of an index, consisting of term location and frequency information. It introduces the inverted index, a representation of terms by the documents which contain them, as an efficient data structure that reduces the time required to conduct a search. It also describes the stemming algorithm, a simple linguistic tool which truncates words to their root form in order to reduce the size

of the vocabulary. It concludes by outlining the framework of the package for text retrieval that will be used in this work, known as the text database (TDB) system.

The third chapter introduces the basic information retrieval strategies, the Boolean Model and the Vector Space Model. The Boolean Model consists of a series of logical constraints applied to query terms to isolate a subset of documents likely to be relevant. The Vector Space Model represents query and document by vectors of word frequencies and estimates document relevance by the inner product of the corresponding word vectors. Relevance feedback, an iterative strategy for improving the quality of the search using relevance judgements provided by the user, and the routing problem are also described in some detail.

The fourth chapter details the problems of feature selection and dimension reduction in information retrieval. It explains why words are such poor features and characterizes many of the previous attempts to deal with this problem. It then presents an alternative dimension reduction strategy known as Latent Semantic Indexing, which applies a matrix decomposition to the term-document matrix to allow both terms and documents to be represented in a lower dimensional subspace of orthogonal indexing variables. The advantages and disadvantages of this approach are considered.

The fifth chapter provides a comprehensive review of methods of evaluation of information retrieval experiments. It describes the problems associated with computer experiments on test collections and explains why the alternatives do not represent a better approach. It characterizes the basic performance measures in IR and gives a basic feel for the intuition behind each measure. It continues by presenting a series of alternative measures and discusses the reasons for their development. It concludes by reviewing the important statistical hypothesis tests which can be used to determine whether one retrieval strategy works better than another in a simulation experiment.

The sixth chapter introduces the classification techniques which can be applied to the information retrieval problem. The primary approach presented is discriminant analysis, which models each group using a multivariate normal distribution and estimates the class membership for new observations by the group which they are

most likely to belong to. The chapter also explains logistic regression, which estimates the log ratio of probability of class membership by a linear model. Several other techniques that proved to be less successful in practice are described, including classification trees and a method for determining the optimal separating hyperplane between two groups.

The seventh chapter presents a series of retrieval experiments on the Cranfield test collection. It begins by providing some summary statistics about the collection and goes on to discuss how to implement the Vector Space Model. It applies the Vector Space Model to the three major search problems in information retrieval to determine a baseline for comparison with other models. Latent Semantic Indexing is introduced to obtain a low dimensional representation of the collection, and the impact of this technique on retrieval performance is measured. Initial attempts are made to apply statistical classification to the routing problem. We find that discriminant analysis applied to a local query-specific representation yields a significant improvement in performance.

The eighth chapter continues our work on retrieval experiments on a much larger collection of documents from the Wall Street Journal. It quickly becomes apparent that Latent Semantic Indexing cannot be applied successfully to the full collection. A new strategy of obtaining a sample of documents in the region of the query is introduced, for which dimension reduction provides a large improvement in performance. Several classification methods are then applied, but they do not prove to be markedly more effective than a simple strategy using the centroid of the relevant documents. The final chapter presents our conclusions and makes some suggestions for future work.

Chapter 2

The Basics of the IR System

For the work in this study, the most important aspect of the IR system is the search engine, which uses a model based on word frequency information to identify relevant documents. However, search tools are built on top of layers of system components that perform the basic tasks of text analysis. Before examining the theoretical IR problem, it is important to describe the process by which the computer converts text into a form that can more easily be stored and processed.

For most IR systems, the basic unit of text analysis is the word. It is the smallest unit of language that carries a reasonable amount of semantic structure. While phrases or sentences may be more meaningful, one must perform an extensive amount of linguistic analysis to capture important structure at this level. Therefore, most text retrieval systems measure the relationship between documents primarily on the basis of word frequency information. Computers can estimate document similarity using word co-occurrence information quickly and efficiently, and this forms the basis for the standard information retrieval system.

This chapter will cover the basic components of a retrieval system which convert raw text into an organized representation as index terms. These tools are applicable to almost any retrieval strategy and can be broken down into two parts, text processing and automatic indexing. Text processing converts the text into a stream of tokens, consisting primarily of words, but also including numbers, abbreviations, and alphanumeric sequences. Automatic indexing describes the process by which the system

stores these tokens in a data structure in an efficient manner so that they can be easily accessed by the search engine. At the end of the chapter, we will provide some details about the retrieval system used in this thesis, known as the Text Database (TDB) system [16].

2.1 The Inverted Index

After a document is processed, the words that it contains can be stored as a list of index terms and their frequencies. While document profiles are easy to construct, they are not the most useful data structure for retrieval purposes. An alternative representation, known as the inverted index, stores a description of each index term which consists of the reference numbers of all documents which contain that term. The inverted index is a crucial part of almost any retrieval system because it improves search efficiency in a dramatic fashion.

Many queries consist of only a small number of index terms and searching every document for these terms would be extremely inefficient. By using the inverted index, the system can quickly identify which documents contain query terms and thus reduce the search space substantially. As collections grow in size, increasing search times, this becomes an extremely important advantage. The one drawback of the inverted index is that it often requires a considerable amount of storage, sometimes enough to double the space required for the collection [23].

However, the difference in performance depends on the size and frequency of the query terms. Long queries with frequent terms will cover almost every document in the collection, leading to no reduction in the size of the search. A number of strategies have been proposed for improving efficiency for long queries and large text collections. A simple and common approach is to eliminate a small number of the highest frequency terms, which are collectively known as the “stop list”, that are likely to have little value in retrieval. This process will be described in the next section.

Alternatively, the system can sort the inverted index entries for each term in order of their frequency and only process the entries until a threshold score is reached. Any

matches below the threshold are assumed not to contribute significantly to the document's score. Persin [60] uses one threshold to determine when to insert a document into the scoring list and a second threshold to determine when to add to the score of a previously included document. His system is able to perform the retrieval task in roughly one fifth the usual time with almost no loss in accuracy. The threshold constants do need to be carefully tuned to optimize performance, but the potential for improvement is dramatic.

The amount and type of information stored in the inverted file posting for each index term can depend on the search application. If the search technique does not require term frequencies, the term description can be merely a list of documents. If documents will be compared to the query on the basis of a frequency-weighted similarity measure, the postings must also include the frequency of the term within each document. Finally, if the goal is to duplicate the structure of the original text (perhaps excluding the changes described in the next section), the system can store positional information for each term in the indexed document [17].

For any text collection of reasonable size, the inverted index is stored in a data structure on disk, such as a sorted array or a B-tree [17]. A sorted array can be accessed using binary search and is easy to implement but relatively hard to update. Frakes [23] presents a detailed implementation. A B-tree is a balanced multiway search tree that provides efficient access to secondary memory for large databases. Entries can be found in $O(\log_b N)$ disk accesses where N is the number of entries and b the branching factor of the B-tree. Since b can be relatively large, entries can be accessed extremely efficiently [17]. B-trees require more storage than a sorted array, but are much easier to update.

2.2 Lexical Analysis

The system can convert raw text into tokens in a relatively straightforward manner. There are some issues to be addressed, such as how to handle punctuation (hyphens), how to find sentence, paragraph, section, and document boundaries, whether and when to distinguish between upper and lower case, etc . . . While these issues can

have an impact on retrieval performance, they can usually be resolved in a relatively simple fashion. For example, proper names are generally identified by a connected sequence of capitalized words and can be very helpful in retrieval. Instead, we will focus our attention on two other important components of the text processing system, stop lists and stemming algorithms, which deserve more comprehensive study.

2.2.1 The Stop List

There exists a large class of words that have no inherent meaning when taken out of context and thus are basically useless as index terms. For example, terms such as “the”, “and”, or “of” have no semantic content. These words also tend to be among the most frequently occurring terms in the English language. Therefore, it seems natural to filter these terms out of the inverted index. This is done by creating a list of terms that will not be indexed, otherwise known as a stop list.

Using a stop list of function words can significantly increase the efficiency of a retrieval system by reducing both the size of the inverted index and the time required to conduct a search. A small number of very frequent terms often accounts for as much as a third of the tokens in a document collection. Therefore, removing them reduces the size of the inverted index proportionally. The most common stop words are likely to occur in almost every document in the collection. If these terms are present in the query, then most of the document collection will have to be examined to conduct a search.

Opinion varies as to the optimal number of terms on the stop list. Stop lists can range in size from perhaps eight to as many as 500 individual terms. Modern approaches have tended to favor stop list towards the larger end of that range, although the size also depends on the search method. The contents of the stop list may also be domain dependent, as certain index term that are uncommon in general use may be exceedingly common in a specific context, which might merit their removal. One should keep in mind that the stop list should not be selected solely on the basis of frequency, for some content-bearing words are very frequent in the English language. Stop words can also provide other clues, such as information about the genre of a

document. For example, the biographical entries in Grolier's encyclopedia [1] have an extremely large number of pronouns. Whether this information can be used constructively for information retrieval remains an open question. It has been found in retrieval experiments that using a stop list in the size range listed above does not make it harder for search algorithms to identify relevant documents [23].

2.2.2 Stemming Algorithms

Another common strategy for reducing index size and potentially improving retrieval performance is to apply a stemming algorithm to word tokens. A stemming algorithm is a linguistic tool for building word equivalence classes for index terms by removing and modifying prefixes and suffixes to identify the root form of the word. All variants which reduce to the same root stem are combined to create a single index term. This strategy is based on the assumption that most common morphological variants of a word have similar meanings and are used in similar contexts. For example, if a query contains the word *colonization*, the user is also likely to be interested in *colonize*, *colonized*, *colonizing*, *colonist*, and even *colony*. If all of these variants can be reduced to a their simple root form, such as *colony*, this should help to extend the coverage of the search. Using a stemming algorithm will also reduce the size of the inverted index, often by as much as 20-50% for a small collection. This reduction is not as substantial for larger, more general document collections [32].

There have been a large number of experiments conducted to test the performance of stemming algorithms [23], with mixed results. Some studies have found that stemmers do improve retrieval performance while others have found no difference between systems running on stemmed and unstemmed text. In general, it appears that stemmers do not degrade retrieval performance, and the specific choice of stemmer from among the most popular variants does not seem to be particularly important. The effect of stemming may well depend of the nature of the vocabulary used in the collection.

When might stemming a word not be productive? Sometimes different forms of the word may be used in different contexts, and thus deserve separate entries in the

index. For example, the term “stabilizer” may be used to refer to an aircraft part and may not be used in the same context as stabilize, stable, or stability. Gale et. al. [28] find an average correlation of .67 between documents that use singular and plural forms, and for the lower 70% of the terms the correlation is .37, which indicates that the plurals for some terms are probably used in different contexts.

A number of different stemming algorithms have been proposed. Most stemmers work by iteratively applying a set of rules for prefix and suffix removal. A simple rule of this form is one which converts the plural to the singular form. An example of a more complex rule follows [32]: “If a word ends in “ies” but not “eies” or “aies”, then “ies” $\Rightarrow$ “y””. Additional constraints are added to handle as many exceptions as possible.

2.3 The Text Database System

The basis for the retrieval system used in this thesis is the Text Database (TDB) System developed by Jan Pedersen and Doug Cutting at Xerox PARC [16]. The TDB System, written in Common Lisp [43], is an object-oriented prototype designed to provide a flexible framework for experimental work in information retrieval. The system is decomposed into a set of modules that allows considerable variety in the range of implementations at different levels. The important components of the system that will be used in this work are the following modules: Corpus, Analysis, Indexing, and Storage.

The Corpus module produces a mapping from a document to the text which it contains. It removes control characters and identifies document boundaries. The document text is represented as a character stream which can be passed on to other applications. Once a few basic corpus templates have been developed, new document collections can easily be added to the system.

The Analysis module converts the text into terms suitable for indexing. First, the character stream is converted to a token stream, where a token is defined as a sequence of alphabetic and/or numeric characters. Subsequent steps are accomplished by a series of filters. Stop list tokens are removed and the remaining tokens are passed

to a stemming algorithm.

The Indexing module creates index structures useful for applying various search strategies. The inverted index allows document location and frequency information to be quickly obtained for any index term, as described in the first section. The system can also access document profiles which describe each document by its term frequencies. The Storage module uses B-trees to store document profiles and the inverted index on disk.

Using the tools from the TDB system, one can quickly and easily access descriptions of either terms or documents. The document collection can now be studied with a wide variety of search tools. The search and retrieval methodology, which forms the heart of the IR system, will be described in detail in the next chapter. Specific details of the implementation of these search strategies will be given in chapters 7 and 8. The computer code for all the basic search strategies used in this thesis was written by the author in Common Lisp. The classification algorithms that will be described in chapter 6 were run using SPlus functions and computer packages which are freely distributed by a variety of sources.

Chapter 3

Search Strategies in Information Retrieval

We now address the fundamental problem in information retrieval. Given a query, how can the system automatically determine which documents are most likely to be relevant? In the previous chapter, we described how to index raw text so that the characteristics of both index terms and documents can be easily accessed. For the most part, the search strategies described here will rely on the word as the basic component of the system, since this allows us to make use of the efficient term retrieval tools developed in the previous chapter. Since our system will perform no natural language processing or linguistic analysis other than using a stemming algorithm, the only objects used to measure the relationship between the query and documents will be the index terms that they have in common. The fundamental assumption of these models will be that the more vocabulary shared by the document and the query, the greater the likelihood that the document is relevant to the query.

This chapter will examine two of the basic models used in information retrieval, the Boolean Model and the Vector Space Model. The Boolean Model retrieves documents that satisfy a series of logical constraints on the index terms in the query. While fast and easy to implement, it has a number of problems that often make it an ineffective retrieval tool. The Vector Space Model treats queries and documents as vectors of term weights and computes an inner product similarity score between them. It is

more flexible than the Boolean Model because documents are ranked in order of their estimated relevance, and term importance weighting strategies can be applied. It also is ideally suited for automatic relevance feedback, where the system interactively updates the query based on relevance judgements provided by the user. The chapter will conclude by providing a brief discussion of the routing problem.

3.1 Boolean Retrieval

Boolean search represents a simple generalization of the database look-up strategy described above. In Boolean retrieval, the system selects those documents that satisfy a logical expression of query terms using connective expressions such as OR, AND, or NOT. In mathematical terms, OR = union, AND = intersection, and NOT = complement. A query might be expressed in the form $Q = (A \text{ or } B) \text{ and } C$, which corresponds to: find all documents which contain term C and either term A or term B. Boolean search can also be accomplished in an efficient manner. The system looks up the entry for each term in the inverted index and merges the resulting document lists, following the constraints of the boolean query expression. Most commercial retrieval systems, such as library catalogs, still rely on this search strategy.

3.1.1 Drawbacks of Boolean Model

Boolean retrieval suffers from a number of serious drawbacks summarized below [64]:

- (1) No control over the size of the output.
- (2) Not robust; query extremely sensitive to the choice of terms.
- (3) No concept of term importance.
- (4) No ranking of document importance.
- (5) Difficult to formulate and modify complex queries.

In the following paragraphs, we analyze each of the drawbacks in more detail and describe methods that have been proposed to overcome them.

The number of documents satisfying a boolean query is highly variable, depending on the frequency of the query terms and the number of conjunctions/disjunctions in the query expression. The union of frequent terms can result in far more documents than the user could possibly examine while the intersection of rare terms can result in no documents being retrieved at all. A user could conceivably adjust the query by adding conjunctions to increase the size of the retrieved set or disjunctions to decrease it, but this is very imprecise at best, because of the user's lack of knowledge about term frequency and overlap.

In addition, the query is extremely sensitive to the choice of retrieval terms and the logical connectives used to link them. Adding a single term to the query can change the output substantially. This is not a desirable property if the user is not sure of exactly what they are looking for. Using a large number of disjunctions forces the document to use exactly the same vocabulary as the query, which makes the search method inflexible to slight changes of vocabulary.

The boolean query includes no concept of term importance. Therefore all terms are assumed to be equally important in practice, which severely limits the user's ability to adjust the query in a subtle manner. Similarly, all documents returned by the user are assumed to have equal importance. If the output is large, the user has no idea if a particular document matched on only a single relatively useless term, or on most or all the terms in the query. Without knowing which documents are likely to be more important, the user may have to spend a considerable amount of extra time analyzing the search results.

Finding the correct logical combination of query terms is an elusive goal. In particular, as the query gets more complex, the number of possible combinations grows extremely rapidly. It takes a lot of time and effort to design and tune such queries. Furthermore, if a relevant document is found, it can be difficult to figure out how to use that information to improve the query. In particular, there is no automatic way for the system to use a relevant document to improve the quality of the search. Thus, a considerable amount of human effort is required to construct a complex and accurate boolean query.

3.1.2 Improving the Boolean Model

Several methods have been proposed for dealing with the drawbacks associated with the Boolean Model. InfoCrystal [74] is a visualization tool that allows the user to view a graphical representation of all logical combinations of query terms simultaneously. The user can use this display to identify how many documents exist for each combination, which can help with many of the problems described above. The user can see the size of the output and select a boolean combination of query terms which has a reasonable number of documents. The graphical display makes it possible to see which query terms are most useful for increasing or decreasing the search output, and the user is no longer burdened with selecting the optimal boolean relationship between query terms before seeing how they are distributed. However, since the number of combinations grows exponentially with the number of query terms, this representation is useful only for a small number of query terms.

For systems with a controlled vocabulary of common index terms, the system can provide help in narrowing or broadening a search by using a structured dictionary [79]. This dictionary is constructed in advance and stores more general or more precise related keywords for each potential index term. The user can then select from among these terms to change the scope of the search. While such a tool would be extremely valuable, the difficulties associated with its construction limit it to a relatively small vocabulary of index terms.

The Boolean Model can be substantially improved by adding a method for weighting the relative importance of retrieved documents. One simple method would be to rank documents in order of the number of query terms (separated by the AND relationship) that match terms in the document. For example, the query $Q = A \text{ and } (B \text{ or } C)$ could score documents as follows.

doc	A	B	C	score
D1	Y	Y	Y	2
D2	Y	Y	N	2
D3	Y	N	Y	2
D4	N	Y	Y	1
D5	N	N	Y	1
D6	N	Y	N	1
D7	Y	N	N	1
D8	N	N	N	0

Essentially, documents are graded according to the number of times they match a term in the query. With this system, the user can identify the documents with the highest level of partial matching to the query.

This approach has been generalized to produce the extended boolean retrieval model [64]. This model incorporates term weights given by the user and assigns higher scores to cases D1 and D4 to provide a more continuous ranking of documents. The extended Boolean Model addresses most of the problems associated with the uncontrolled size of the output and the lack of term and document weights, which should make the boolean approach more effective.

It is also important to recognize that terms which appear in the same document are not necessarily related in the fashion intended by the boolean query, and may just occur together by chance. This problem is particularly associated with long documents that may cover a wide variety of topics. To attack this problem, many systems will allow the user to impose a proximity constraint on the terms in the query. Not only must they occur together in the same document, but they must have fewer than k words between them. Proximity constraints make sure that the index terms are at least used in the same part of the document.

It is important to note that none of these methods addresses problem (5) above. The Boolean Model is designed to deal with short queries, and it will never be easy to construct and apply very long and complex queries. There is no obvious way to automatically incorporate information from relevant documents into a revised query. This technique, known as relevance feedback, represents one of the important advantages

of the Vector Space Model, which is described in the next section.

3.2 The Vector Space Model

The Vector Space Model (VSM) represents queries and documents as vectors of term weights [67]. Each element of the document vector measures the importance of a specific term in the document, i.e.

$$Q_i = (q_{i1} \dots q_{in}) \quad , \quad D_j = (d_{j1} \dots d_{jn})$$

where q_{ik} and d_{jk} give the value of term k in query Q_i and document D_j respectively. The value of these components can be binary (reflecting the presence or absence of the term in the document), the frequency of the term in the document, or some more complex importance weight assigned to that term.

If terms are assumed to be independent, one can estimate the similarity between a query and a document by the inner product of their corresponding vectors.

$$\text{Sim}(Q_i, D_j) = \sum_k q_{ik} * d_{jk}$$

Clearly, there are strong correlations in the structure of term usage across language, but the independence assumption is probably a reasonable first-order approximation. Essentially, we can consider the inner product score to be a heuristic estimate of the similarity between a query and a document which has the desirable property that the more common vocabulary that exists between the query and the document, the greater the similarity score between them.

3.2.1 Using the VSM for retrieval

Using the Vector Space Model, the system can compute a similarity score for each document in the collection and rank the documents in decreasing order of their similarity score. The highest ranking documents are returned to the user as the result of the search. In practice, the similarity score is not computed sequentially for each document in the collection. Rather, the inverted index is used to identify documents

which have some overlap with the query, and the component for that term is then added to the score of all overlapping documents. This method reduces the number of disk accesses and makes the search go significantly faster.

The Vector Space Model does not suffer from the defects associated with the Boolean Model. Documents are ranked on a continuous scale, so it is easy to control the size of the output and the higher ranking documents are more likely to be relevant. Term weights are incorporated into the model to distinguish which terms are most important. The query is much less sensitive to the choice of particular index terms since evidence is combined in an additive fashion to determine the final ranking. The model deals easily with both short and long queries, since any form of free text can be used as a query. Therefore, one can attempt to improve the query by incorporating some or all the text from a known relevant document directly in the query. The ability to deal easily with free text gives the Vector Space Model a substantial edge over the Boolean Model. Furthermore, it has been shown that the VSM performs better than the Boolean Model in a large number of retrieval experiments.

The one significant problem with the VSM is that it cannot capture the concept of term dependence. The Vector Space Model assumes that all terms are independent so that they can be used as the basis of the vector space. However, it is self-evident that terms have strong dependence patterns. A set of terms used to describe concepts in Chemistry are more likely to occur together than they are to occur with vocabulary from Psychology or Linguistics.

One of the most significant forms of term dependence occurs in two-term phrases. For example, the phrase “United States” is so common in Grolier’s Encyclopedia [1] that “united” and “states” are highly correlated. A boolean query can capture this relationship precisely by requiring that both terms occur in the matching document. The Vector Space Model may assign a high weight to both of these terms, but it is quite possible that a document that uses “states” very frequently could receive an equally high similarity score. Methods for dealing with term dependence will be discussed in subsequent sections.

3.2.2 Term Weighting

One of the important advantages of the Vector Space Model is its ability to use term weights to capture the fact that some terms are far more important than others. While such weights could be selected by the user, this would require a considerable amount of knowledge and effort, so the best solution is to have the system select these weights automatically. Term weights can be broken up into three components: (1) term frequency weights, (2) term importance weights, and (3) document normalization weights.

The simplest implementation of the VSM uses binary weights to indicate whether or not a term occurs in a document. However, by ignoring the frequency of terms in a document, the system loses a lot of useful information. A term that occurs frequently within a document will probably be strongly related to the content of a document while a term mentioned once in passing may have minimal importance. The system can capture this relationship by weighting terms in both queries and documents according to their frequency. Empirical studies have shown that a term's importance generally does not increase linearly with its frequency. Therefore, it is common to transform term frequency by using the log, square-root, or other similar transformation.

However, a term's importance is not related to its frequency within a document alone, but also depends on its frequency over the collection. High frequency terms are likely to occur in large number of documents and thus may be relatively poor at marking the distinguishing characteristics of relevant documents. On the other hand, index terms that appear in only a few documents have a high degree of specificity, and thus are much more likely to be valuable. To incorporate this factor into the model, terms are assigned a weighting component that is inversely related to the number of documents that contain them.

We can turn to information theory to obtain a useful measure of term specificity [83]. Shannon's entropy:

$$H(P_i) = - \sum_{j=1}^N (p_{ij} * \log(p_{ij})) \quad , \quad p_{ij} = 0 \rightarrow p_{ij} * \log(p_{ij}) = 0$$

where p_{ij} is the probability that an occurrence of term i is in document j and N is the number of documents in the collection. Entropy can be interpreted as a measure of the degree of uncertainty associated with the evidence obtained from term i . Entropy is maximized when term i is uniformly distributed over the document collection ($H(P_i) = -\sum_{j=1}^N \frac{1}{N} \log(\frac{1}{N}) = \log(N)$) and minimized when $p_{ij} = 1$ for a particular document ($H(P_i) = 0$). One can measure relative uncertainty by $\beta(P_i) = H(P_i)/\log(N)$.

Relative entropy can be converted to a measure of certainty or specificity by using $w_i = 1 - H(P_i)/\log(N)$. This component is known as the signal to noise ratio and has often been used in retrieval experiments. If w_i is simplified by assuming that there is a uniform probability of term occurrence in each document then $H(P_i) = -\sum_{j=1}^{n_i} \frac{1}{n_i} \log(\frac{1}{n_i}) = \log(n_i)$, where n_i is the number of documents containing the term, and $w_i = 1 - \log(n_i)/\log(N) = c * \log(N/n_i)$. The term $\log(N/n_i)$ is known as the inverse document frequency. The inverse document frequency is the most commonly used term-importance weighting strategy.

The inner product measure of similarity is not normalized in any way to reflect vector length. This means that longer documents will tend to have higher similarity scores simply because they will have more terms and higher term frequencies. This can be balanced by applying a normalization weight to account for document length. This is often done by normalizing the document vector to have unit length, or by dividing by a function of document length. The query does not need to be normalized since all documents will be ranked with respect to the same query and only the ranking of documents is important.

The three weighting components described above are generally applied in a multiplicative fashion to produce term weights that appear as follows:

$$w_{ij} = \frac{f_1(d_{ij}) \log(N/n_i)}{f_2(|D_j|)}$$

where d_{ij} is the frequency of term i in document j and $|D_j|$ is the length of document j . The choice of weighting functions f_1 and f_2 has been the subject of a large number of retrieval experiments. It has been shown that using term weights is significantly better than binary term vectors, and that inverse document frequency weights

substantially improve performance. For other factors, it depends on the collection. Collections with widely varying document lengths benefit from document normalization, and collections with longer queries and documents benefit from within document frequency weights.

3.2.3 Subtopics and Passage Retrieval

In the early days of IR, when computing power was limited, retrieval experiments were run on test collections which were made up of journal abstracts or other short articles. In documents of this size, content is generally limited to a single coherent topic. However, most modern collections contain much longer “full text” documents that will often cover a large number of different topics. The vector space model was designed and optimized on collections of short documents, so it is very reasonable to assume that a global document similarity match may not be the best approach for heterogenous full text documents.

Just as the concept of proximity was introduced for boolean search, one can consider producing a similarity score for subsets of a long document. If the document is divided up into smaller passages, which could be paragraphs, sections, groups of sentences, or even fixed-length windows of terms, a separate similarity score could be computed for each passage, and the passage-level scores might prove to be a better indicator of relevance than document similarity. Several recent experiments have addressed this issue.

Hearst [37] proposes a method for partitioning a full text document into multi-paragraph units selected according to their coherence as a single topic. The algorithm works by computing the similarity (as for query-document similarity described above) between adjacent overlapping text blocks (3-5 sentences long). A plot of these scores over the length of the document reveals a series of peaks and valleys, where the peaks correspond to subtopics and the valleys identify boundaries between subtopics. The document can then be divided into passages which correspond to coherent subtopics. Other researchers have used paragraphs [9] or overlapping windows of a fixed number of words [11].

The VSM can be used to compute the similarity between the query and each passage using term-frequency vectors. The system must then combine this information into a unique document score. The system could use only the top-ranked passage score for each document, or some combination of the highest ranking passages. Buckley et. al. [9] and Callan [11] find that performance is optimized by using only the top ranked passage, while Hearst [36] gets better performance by combining the passages that exceed a similarity threshold. Evidence suggests that retrieval on passage scores alone can work for long texts, but that a combination of document and passage level evidence may provide the best overall performance. In general, using passage-level evidence improves performance by up to 20% for collections of long documents and possibly as much as 5-10% for shorter full text documents, when performance is measured using average precision at fixed recall levels. For more information on evaluation, see chapter 5.

3.3 Relevance Feedback

Query-based information retrieval is a sequential process. It is rare that the initial search retrieves exactly what the user is looking for, therefore he or she can generally benefit from submitting a revised query based on what has been learned about the topic from the initial set of relevant documents. This process is known as relevance feedback [63]. One can modify the query by adding or deleting search terms and/or by modifying term weights. Clearly, manual query construction and revision is a relatively arduous and complex task. Given a relevant document, it is difficult to determine what terms to add to the query and which terms are more or less important. Therefore, the most desirable approach is to develop an automatic relevance feedback process so that the system does the work of revising the query.

An automatic relevance feedback system can be designed which works in the following fashion. The system processes the initial query and returns a list of documents ranked in order of their predicted relevance to the user. The user examines a few of the highest ranking documents, determines whether or not they are relevant, and sends this information back to the system. The system uses the analyzed documents

to automatically construct a revised query and produces a new ranking of the remaining documents in the collection. The user can examine more documents and repeat the relevance feedback process as often as desired. This approach is ideal because user effort is minimized. Users need only determine whether a document is relevant or not relevant and the system does the rest of the work.

3.3.1 Feedback using the VSM

The Vector Space Model is ideally suited for automatic relevance feedback since it accepts free text input for the query. Therefore, one can simply incorporate some or all of the text from the relevant documents directly into the query. Non-relevant documents can be utilized in a similar fashion, although the evidence that they provide must be used in a negative fashion. One of the more successful relevance feedback strategies was developed by Rocchio [63], and works as follows:

$$Q_{rfb} = Q_0 + \alpha \sum_{i \in rel} \frac{D_i}{|D_i|} - \beta \sum_{j \in nrel} \frac{D_j}{|D_j|}$$

This strategy simply adds a weighted sum of the relevant document vectors and subtracts a weighted sum of the non-relevant documents from the query. Relevance feedback using this strategy produces a very large improvement in retrieval performance (from 20-80% or more, depending on the collection) [69, 33, 2].

Adding the full document text directly into the query does have its drawbacks. The number of terms in the query grows rapidly with the addition of evaluated documents, causing searches to take longer and longer, as the performance benefits of the inverted index are lost. A full search must be repeated for each iteration of relevance feedback. Since relevance feedback is supposed to be an interactive process, the system must be able to return feedback results in a relatively short period of time.

3.3.2 Improving Feedback Performance

There are several different parameters that can be optimized in Rocchio's relevance feedback model. First, what are the optimal values for α and β ? Second, how many documents should the user examine before performing the next iteration of relevance

feedback? Aalbersberg [2] examines both of these questions in detail, experimenting on a number of different test collections. He finds that performance is optimized for α between .5 and 1 and for β around .1. However, there is very little difference in performance as long as α is considerably larger than β .

Further experiments demonstrate that performance improves as the number of documents examined between iterations of feedback decreases, and it is optimized by using feedback after every document is evaluated. This is not surprising, since each new bit of information should benefit the retrieval system. However, this must be balanced by the cost of doing a feedback search. The user may not be willing to wait for the system to rerank all the documents after each relevance judgement.

Fortunately, one can obtain roughly equivalent performance with considerably less effort. Aalbersberg's results indicate that there is only a minimal loss in performance by ignoring non-relevant documents altogether. This means the feedback need only be used each time a relevant document is identified.

The simplest relevance feedback strategy is to incorporate the full text of relevant documents into the query, however it is unlikely to be the best strategy. As the query gets longer, there is a significant danger of overfitting. Many of the terms within a document may not be characteristic of the underlying topic that is relevant. Therefore, some method of term selection may be in order. Harman [33] finds that adding only the twenty most important terms to the query, where importance is judged using term weights, is far more effective than adding all of the terms from the relevant documents. We will address this issue in more detail in the next chapter.

3.3.3 Assumptions of Relevance Feedback

While relevance feedback of this kind appears to work extremely well in practice, it is based on a number of assumptions that need to be recognized. Relevance Feedback will be most successful when the distribution of relevant documents is unimodal and can best be approximated by using its vector centroid, perhaps adjusting for the query and the centroid of the non-relevant documents. This need not always be true. For example, the query could be satisfied by documents that come from two substantially

different fields that describe the topic with different sets of vocabulary. In this case, relevant documents may be represented by two distinct clusters, and using relevance feedback may make it even more difficult to find the second cluster.

Second, relevance feedback assumes that if a document is relevant, the user wants to find more documents that look as similar to this document as possible. While these documents may also be relevant, it is possible that they will merely duplicate the information contained in the initial relevant document. The user might be better served by looking for documents that are less similar to other relevant documents. Admittedly, this is a fairly subtle distinction, and it is likely to be outweighed by the benefits associated with expanding the query, but it is a hypothesis that current retrieval experiments are incapable of examining. This issue will be examined further in the chapter on evaluation.

3.3.4 Feedback using a Probabilistic Model

There is an alternative method for relevance feedback based on a probabilistic retrieval model [79]. Let document $x = (x_1 \dots x_n)$ be a binary vector indicating the presence or absence of index term x_j in that document. If one could estimate $P(\text{rel}|x)$, one could then use Bayes Decision Rule $\{ P(\text{rel}|x) > P(\text{nonrel}|x) \rightarrow x \text{ is relevant} \}$ to determine which documents are likely to be relevant. From Bayes' Theorem, the decision rule can be rewritten as:

$$P(x|\text{rel})P(\text{rel}) > P(x|\text{nonrel})P(\text{nonrel}) \longrightarrow x \text{ is relevant}$$

The most common approach to estimating $P(x|\text{rel})$ is to assume conditional independence, i.e. $P(x|\text{rel}) = \prod_i P(x_i|\text{rel})$. Write $p_i = P(x_i = 1|\text{rel})$ and $q_i = P(x_i = 1|\text{nonrel})$, then the likelihood functions for our estimators are:

$$P(x|\text{rel}) = \prod_i p_i^{x_i} (1 - p_i)^{(1-x_i)} \quad , \quad P(x|\text{nonrel}) = \prod_i q_i^{x_i} (1 - q_i)^{(1-x_i)}$$

Taking the log likelihoods and plugging them into the decision rule above, we obtain:

$$g(x) = \sum (c_i * x_i) + C_0 \quad , \quad c_i = \log\left(\frac{p_i(1 - q_i)}{q_i(1 - p_i)}\right)$$

The constant C_0 determines the cut-off for relevance. Therefore, if we can estimate p_i and q_i , we have a natural set of term weights for use in retrieval.

Given complete information about which documents are relevant and not relevant, we can define a contingency table measuring the occurrence frequency of term x_i as follows [79]:

	Relevant	Non-relevant	
$x_i = 1$	r	n-r	n
$x_i = 0$	R-r	N-n-R+r	N-n
	R	N-R	N

Using this table, we can define $p_i = \frac{r}{R}$ and $q_i = \frac{n-r}{N-n-R+r}$. Clearly, complete information is never available, so it is common to form estimates based on a partial contingency table. Such a table can be built during relevance feedback.

Since this model does not account for the frequency of a term in a document, some researchers substitute the probabilistic term weight for the inverse document frequency and also include term frequency and document normalization components [33]. This strategy suggests scores for reweighting the current query. It is also often used as a criterion for query expansion. Other terms which have a high score according to this measure are added to the query.

The probabilistic model suffers from much the same flaws as the Vector Space Model; it does not model the dependence structure that is known to exist between terms. While it should suggest reasonable weights when there is enough data to accurately estimate the probabilities, in practice this is rarely the case when relevance feedback is being attempted. It is also based on a binary model of word occurrence and may not generalize well when frequency information is included.

Researchers have attempted to generalize this basic approach by estimating word frequency distributions using a poisson model. In this method, a term would receive a high weight if its frequency over the relevant documents was extremely unlikely to occur given a poisson model of term frequency over the collection. However, the poisson model assumes that the term is uniformly distributed across the collection. Mosteller and Wallace [55] found that the poisson model does not fit very well to

high frequency function words, and subsequent studies have demonstrated that content words are even more likely to be grouped in clusters. Other attempts include modelling term frequencies by the mixture of two or three poisson distributions, but none of these approaches have produced better results in retrieval experiments than the basic vector space model [75, 28].

3.4 The Routing Problem

In the adhoc search problem, the system must rank documents based only on the text contained in the initial query. In the routing problem, it is assumed that a large number of documents have previously been examined, and that information about their relevance status is known. The goal is to predict the relevance of an incoming sample of documents that have not yet been examined.

The traditional approach to this problem has been to use the techniques developed for relevance feedback. Query terms are reweighted according to their importance, as estimated from the evaluated documents, and additional important terms are added. Documents are ranked according to the similarity score computed from the inner product of the document and revised query vectors. Nearly all of the most successful routing methods from the recent TREC-2 conference [35], used some variant of this approach.

From a statistical standpoint, this approach is equivalent to ranking documents according to their Euclidean distance from the weighted centroid of the relevant documents, assuming that documents are normalized to have equal length. While the scores are different, the rankings in order of decreasing similarity score and increasing distances are the same. Clearly, this a very simple approach to the routing problem, since it does not make use of any information associated with the distribution of the relevant and non-relevant documents.

The primary goal of this thesis will be to obtain improved retrieval performance by applying statistical classification techniques to the routing problem. These methods develop more general models of the structure of the relevance space than can be obtained from a simple feedback query, and so they should be able to make more

productive use of the large amount of information available in the routing problem. It is hoped that classification methods may be of some use in improving performance for relevance feedback, although this task is much more challenging because so much less data is available. These methods will be discussed and analyzed in retrieval experiments in chapters 7 and 8.

Chapter 4

Feature Selection and Dimension Reduction

Words are probably the most convenient features for characterizing documents. Most have an inherent meaning and occur with enough frequency that it is feasible to claim that documents are related if they have a large number of terms in common. Unfortunately, word features are probably not the ideal information carrier for query-based information retrieval. This chapter will examine the problems associated with word features and suggest an alternate model, known as Latent Semantic Indexing, which has the potential to be much more successful, particularly when used in conjunction with statistical classification.

4.1 Problems with the VSM

Most successful information retrieval systems, such as those based on the Vector Space Model, rely on term frequencies within a document as the primary source of information. A term-based model suffers from a number of problems that have been addressed by researchers over the years. The most important drawbacks are summarized below:

- (1) Synonymy - more than one term has the same meaning

- (2) Polysemy - a single term has more than one meaning
- (3) Term dependence - many terms carry similar information

This section will describe each of these drawbacks in more detail.

Synonymy (or the use of synonyms) refers to the fact that the same underlying concept can often be described using different terms. Traditional retrieval strategies that are based on term co-occurrence data will have great difficulty finding documents on the same topic as the query that use different vocabulary. This is a particular problem for document collections that span a number of different fields where the same concept is often described in a different manner. For example, the term “cluster analysis” is well known in the statistics community, but this topic is also studied in machine learning where it is commonly referred to as “unsupervised learning”. In order to discover this type of relationship, a retrieval system must look at second order similarity relations, i.e. it must recognize that terms are related not just if they co-occur in the same document, but if they each co-occur with the same terms.

Polysemy describes words that have more than one meaning, which is a common property of language. For example, the term “suit” has a very large number of meanings. It could be a court action, a piece of clothing, a symbol in a deck of cards; if used as a noun, or other definitions if used as a verb. If important polysemous words occur in the query, they can significantly degrade the quality of the search by retrieving documents that are completely unrelated to the topic of interest. A number of researchers have looked at term-disambiguation strategies that can help identify the particular use of a term from its context.

The Vector Space Model assumes that terms can serve as independent basis elements for the vector space. In reality, there are often strong patterns of dependence between terms used to describe similar topics or concepts. The result of this is that the contribution of terms with positive correlation is overestimated in the VSM. The highest correlations are often found in phrases, where the components of the phrase do not often occur separately. Take an extreme example where two words always occur together in a phrase, and never separately. The similarity score is incremented by a product of the term weights for each term in the phrase. The phrase should

probably be considered as a single unit and thus be added to the similarity score once. Therefore, the phrase is contributing two or more times its actual value to the score for that document. Some systems deal with this issue by identifying phrases and indexing them as a combined unit.

Many of the problems described above are not unique to the vector space model, but are underlying difficulties with using term frequencies as the basic model for queries and documents. In addition, there is the fact that each document contains hundreds to thousands of distinct terms and a collection may have half a million. It is a struggle for any model to absorb all of this data. Furthermore, most terms occur in a very small number of documents, so the information which can be obtained from any single term is limited.

4.2 Improving the VSM

The problems with a term based model raise a number of issues relating to variable selection. Do we need more or fewer terms in the query? If the query is short, individual terms might be such poor predictors that the model could benefit from finding additional useful terms. In cases such as routing or relevance feedback, there are an excess of index terms available and it is possible to directly estimate their value, so term selection may well be helpful. Alternatively, one might replace or augment terms with other variables, disambiguated terms, phrases, term clusters, or other alternatives and end up with better performance. Previous research on each of these options will be described in this section.

4.2.1 Query Expansion

The best way to deal with synonymy is to identify additional index terms that describe the same concept given in the query and add them to the query. This is known as query expansion. By adding additional useful terms to the query, we can potentially recover documents that describe the relevant concept using different vocabulary. The key question is how to find those terms. One approach would be to use cluster analysis

to identify clusters of similar terms. One could then add to each term in the query the other terms contained in its cluster.

There has been an extensive amount of research conducted on term clustering. Early research [42, 54] found little benefit to using cluster analysis for query expansion, although it did improve performance slightly for some collections. These experiments suggested that small clusters of relatively rare terms produced the best results. On the other hand, Crouch [15] obtained much better results in a later experiment.

The key difference between these experiments lies in how they compute term similarity. The earlier experiments measure similarity by the cooccurrence of terms within documents while the later experiment first clusters documents and looks for terms that occur in all documents in the document cluster. This is a very important distinction. First order term similarity (term A and term B occur often together) is of limited utility for several reasons. First, the majority of documents identified by the clustered terms will already be highly ranked due to the presence of the original term. Second, many of the synonyms of interest are from different domains and thus will rarely cooccur with the query terms.

Probably the best way to find these hidden synonyms from term clusters is to find that both the term and its synonym occur in documents that have a large number of other terms in common. These second order effects can be captured by looking for synonyms in clusters of similar documents. Unfortunately, it is very time consuming to perform a detailed cluster analysis of large document collections.

An alternative approach is to rely upon an external thesaurus to identify similar terms. Salton and Lesk [65] obtain significant improvement in several collections, but it requires the use of a manual thesaurus. Voorhees [81] performs query expansion using Wordnet, a manually constructed network of lexical relationships, and finds that expansion helps only for very short queries. For longer queries, it can actually hurt performance. Term similarity can also be computed based on linguistic knowledge, which leads to a slight improvement in performance [31].

Query expansion need not be limited to terms that cluster together with query terms. Qiu and Frei [61] obtain a relatively large improvement (15-30%) in performance by adding terms that have the greatest similarity to the entire query, rather

than individual terms. Terms are represented in a term-term similarity matrix, and the query is constructed as the union of the terms that it contains. Unfortunately, construction of the full term-term similarity matrix is also prohibitively difficult for large collections.

In summary, query expansion to obtain additional useful index terms sometimes improves retrieval performance. However, the methods that are successful require a considerable amount of computational effort. It should be mentioned that when relevance information is available (for feedback and routing) query expansion can be extremely successful, as described in the previous chapter.

4.2.2 Word Sense Disambiguation

A classical approach to dealing with polysemy is to perform word sense disambiguation on the ambiguous terms. Many different approaches have been tried, including the use of dictionaries, morphological analysis, bilingual corpora, and Wordnet [70]. While disambiguation of individual terms can often be accomplished with a high degree of precision [84], it has rarely led to improved performance for an IR system and has often produced a significant drop in performance [80].

Sanderson [70] runs a simulation experiment where he replaces groups of words at random with a pseudo-word to introduce artificial ambiguity and then measures the change in performance of the IR system. He finds that one can add a large amount of ambiguity before performance is extensively degraded. These results imply that word sense disambiguation is not particularly useful in information retrieval. If the queries are long enough or of high quality, then one can expect the remainder of the query to provide enough context that ambiguous words do not cause problems.

However, one can easily imagine specific examples where a short query using a rarer sense of a particular term could benefit from disambiguation. For example, Sanderson cites queries looking for “John Major” (British Prime Minister) that matched to a large number of documents where “major” was used as a military rank. One problem is that word sense disambiguation is very difficult to evaluate because the correct sense must be determined by hand for each term disambiguated, so most

disambiguation systems have only demonstrated their performance over a small number of terms on shorter collections.

4.2.3 Phrase Modelling

Term-based retrieval models usually assume term independence, which is probably a poor approximation to reality. Among the most common violators of this assumption are terms that occur in common phrases, since the terms always occur together when the phrase is present. This type of term dependence can be incorporated into the retrieval model by identifying and indexing term phrases.

There are a number of different methods for identifying phrases. Proper noun phrases, like people, places, and organizations, can often be identified simply on the basis of capitalization, but this is clearly a special case. Phrase recognition techniques fall into two major classes: statistical and syntactic. Statistical methods merely identify sequences of terms that occur together much more commonly than would be expected under the term independence model, generally using some measure of correlation such as a mutual information measure. Syntactic methods use the linguistic structure of the text to identify groups of related words.

Fagan [22] conducted an extensive series of experiments comparing methods for statistical and syntactic phrase generation in information retrieval. He found that simply defined statistical phrases worked considerably better than syntactic phrases while both led to some improvement in performance. Lewis and Croft [50] suggest that the poor performance of syntactic phrases may be due to their poor statistical properties over small test collections. By this, they mean that most of the syntactically generated phrases were found too rarely to be useful for retrieval.

For large text collections, indexing phrases is difficult because of the very large number of possible combinations. For this reason, most phrase indexing strategies in this context are limited to frequently occurring word pairs [9].

4.3 Latent Semantic Indexing

We have seen that terms are a reasonably poor representation of the underlying meaning of language. They suffer from ambiguity and redundancy and their dependence structure is very difficult to model, primarily due the number of terms available. Therefore, our goal is to provide an improved representation of documents that will:

- (1) reduce or eliminate synonymy, polysemy, and term dependence
- (2) reduce the dimensionality of the problem

The methods described in the previous section can help with the problems defined in (1), but other than term clustering, they do not reduce the dimensionality of the document representation. In fact, incorporating sense disambiguation and/or phrases into the retrieval system actually increases the dimensionality of the representation.

In this section, we will describe Latent Semantic Indexing (LSI), a technique that will address all of the problems associated with terms and the VSM and lead to substantial dimension reduction. The retrieval data used by the VSM can be represented without loss of information by a document by term matrix, where each entry gives the frequency of a term in the corresponding document. The document by term matrix is simply a very large data matrix for a population of documents, each described by a vector of term features.

In the statistics community, a commonly used data reduction technique is Principal Components Analysis (PCA) [3]. Principal components are linear combinations of the original variables constructed to maximize the variability in the data which can be explained by a limited number of features. The first principal component is the linear combination with maximum variance, and subsequent components share this property with the additional constraint that they must be orthogonal to all previous components. Principal components are extremely useful because one can replace a large set of features with a much smaller set of components and still explain the majority of the variability in the original data.

Principal components are computed from the covariance or correlation matrix of the original data. The component vectors are simply the eigenvectors of the corresponding matrix, as demonstrated below:

$$\max_x \frac{\|(x^\dagger S x)\|}{\|x^\dagger x\|} = \max_x \frac{\|x^\dagger U^\dagger \Lambda U x\|}{\|x^\dagger x\|} = \max_x \frac{\|(y^\dagger \Lambda y)\|}{\|y^\dagger y\|}$$

Where S is the correlation or covariance matrix of data, $S = U^\dagger \Lambda U$ is the spectral decomposition of S into an orthogonal matrix U of eigenvectors and a diagonal matrix Λ of eigenvalues. Let $y = Ux$. $U^\dagger U = I$ since U is orthogonal. Then the maximum value is λ_1 which is attained at $Ux = [1, 0, \dots, 0] \rightarrow x = u_1$, the first eigenvector of S . The second and subsequent components can be derived in the same fashion. The eigenvectors give the coordinates of the new variables and the eigenvalues describe their relative importance, in terms of amount of variability explained. The user generally selects a subset of the largest components to use as the new variables, using some criterion based on the decay rate of the eigenvalues.

4.3.1 Description of LSI

Latent Semantic Indexing [19] is a dimension reduction strategy similar to Principal Components Analysis that is based on the Singular Value Decomposition (SVD). While PCA is based on a symmetric, positive definite matrix, the SVD can be applied to any matrix. It represents a data matrix X as:

$$X = DST^\dagger$$

If X is n by k , D and T are n by n and k by k orthogonal matrices respectively, and S is a diagonal matrix of singular values, with at most k non-zero values. Without loss of generality, we can make all elements of S non-negative and sort them in decreasing order of magnitude. Furthermore, we can delete columns $k+1$ to n from D and reduce S to a k by k diagonal matrix by eliminating rows and columns which have diagonal elements equal to zero.

If X is a document by term matrix, then the rows of D and T represent documents and terms respectively in a new coordinate space defined by the columns of these

matrices. The columns of D and T are the singular vectors of X and are nicely described by Deerwester et al. [19]:

. . . these factors may be thought of as artificial concepts; they represent extracted common meaning components of many different words and documents. Each term or document is then characterized by a vector of weights indicating its strength of association with each of these underlying concepts. [. . .] Our aim is . . . to represent terms, documents, and queries in a way that escapes the unreliability, ambiguity, and redundancy of individual terms as descriptors.

The power of this decomposition lies in its potential for dimension reduction. The SVD is strongly related to PCA, since D contains the eigenvectors of $XX^\dagger$, T contains the eigenvectors of $X^\dagger X$, and the singular values are simply the square root of the eigenvalues of the corresponding matrix product. Note that the matrix X is not normalized by subtracting the mean for IR applications because it is very important to retain the sparseness of the original data for computational reasons. To reduce the dimension of the data, we can simply delete a number of the least important singular vectors (LSI factors). This is accomplished by deleting columns from the right hand side of D and T and reducing the dimension of S in a corresponding fashion. The truncated data matrix maximizes the information retained from X in a lower dimension. This representation will prove to be extremely useful for solving information retrieval problems.

The document by term matrix has several unique properties when compared to the typical matrix analyzed with the SVD. First, for most IR problems it is immense, covering many thousands of terms and documents. This makes computing the SVD an extremely challenging problem, since the standard methods are of $O(n^2k)$ where $n > k$ and n and k will both be in the thousands [30]. Fortunately, the matrix is also very sparse, since most terms only occur in a small number of documents. There are special techniques for solve the large sparse eigenvalue problem which can reduce the computation to $O(nk)$. The one that we will employ in our experiments is known as the single vector Lanczos method, as found in SVDPACKC [21] which is

freely available through NETLIB, a repository of code for solving common problems in numerical analysis.

4.3.2 Benefits of Using LSI

LSI addresses most of the major problems associated with using terms as the features in a retrieval system. Since LSI captures the important correlational structure in terms and documents, it will automatically recognize synonyms and if they are used in similar contexts, both will receive high weights in the same LSI factor. Therefore, one can think of LSI as performing automatic query expansion without increasing the dimension of the corresponding output. Its effect on polysemy is difficult to resolve, but it is easy to imagine that the problem will be less severe due to the additional context that is automatically brought to the query by the use of LSI factors.

The use of factors reduces the term dependence problems associated with the Vector Space Model. LSI factors are orthogonal by definition, and terms are incorporated into the LSI factors in a way that reflects the correlation in their use across documents. Furthermore, term associations are modeled in a way that does not increase the dimension of the problem, unlike methods which index on term pairs or phrases. Finally, Latent Semantic Indexing performs a very large amount of dimension reduction, which means that query and documents can be represented by a relatively small number of important feature variables. This will be crucial when applying statistical classification techniques which are very prone to overfitting the training data.

4.3.3 Drawbacks of Using LSI

Using LSI adds a significant amount of storage and computational cost to the search application. While the reduced document representation might appear to cut storage costs substantially, it is the opposite which actually happens. For example, the document by term matrix for the Cranfield collection which is used in some of our experiments has 90,441 non-zero entries (after stemming and stop word removal). Retaining only 100 LSI vectors over 1397 documents requires storing 139,700 values for the documents alone. The term vectors require the storage of hundreds of thousands

of additional values. In addition, the LSI values are real numbers while the original term frequencies are integers, adding to the storage costs. Using LSI vectors, we can no longer take advantage of the sparse nature of the term by document matrix. With recent advances in electronic storage media, the storage requirements of LSI are not a critical problem, but the loss of sparseness has other, more serious implications.

One of the most important speed-ups in vector space search comes from using an inverted index. As a consequence, only documents that have some terms in common with the query must be examined during the search. With LSI, however, the query must be compared to every document in the collection. Fortunately, several factors can reduce or eliminate this drawback. If the query has more terms than its representation in the LSI vector space, then inner product similarity scores will take more time to compute in term space. For example, if relevance feedback is conducted using the full text of the relevant documents, the number of terms in the query is likely to grow to be many times the number of LSI vectors, leading to a corresponding increase in search time. While data structures such as the k-d Tree [25] exist which can speed the search for nearest neighbors, they are most effective in lower dimensions and only provide a partial ordering of the document collection. However, other techniques, such as query expansion, suffer just as heavily from the problems described above, and LSI performs relatively well for long documents due to the small number of context vectors used to describe each document. These computational and storage costs must be weighed against the advantages of the LSI decomposition.

4.3.4 Using LSI for Information Retrieval

It is relatively simple to use the LSI based document representation for retrieval. LSI can take advantage of the known benefits of applying term weighting in the Vector Space Model. Rather than using raw term frequencies as components, the matrix can be transformed by applying the weighting function from the previous chapter. The SVD of the term-document matrix is then computed and a set of m singular vectors is retained to form the term and document representations. Terms and documents are represented in the same space and the document vectors are simply the weighted

combination of the vectors for the terms that make up that document.

As with Principal Components Analysis, it is difficult to know exactly how many dimensions to retain. For most statistical problems, the “best” dimension is chosen by examining the rate of decay of the eigenvalues and looking for sharp drops in magnitude. Unfortunately, for most IR applications, the singular values decay smoothly and it is difficult to guess at an optimal dimension. Fortunately, we can use retrieval experiments to examine the effect of using different numbers of LSI factors. In general performance increases rapidly to a maximum and then levels off or declines when more factors are added. The optimal number of factors for retrieval depends on the size of collection, but generally is at least 100 for collections that have more than 1000 documents.

In order for information retrieval to work, the query must be represented in the same space as the documents. The LSI representation of documents has the important property that each document vector can be derived by taking the weighted combination of its LSI term vectors, where the weights come from the original term-based representation of the document. Using the same technique, we can construct the query by computing the weighted sum of the LSI vectors of the terms that it contains. Similarity between documents and the query can then be computed as the weighted inner product of corresponding term and document vectors:

$$sim(Q, D) = \sum_i q_i * s_i * d_i$$

where q_i and d_i are the weights for the i 'th LSI factor for the query and document respectively, and s_i is the i 'th singular value. Documents are then ranked according to their similarity score and a subset of the highest ranking documents is returned to the user, the same strategy used for term-based representation of documents.

Additional terms and/or documents can be folded into the LSI space of the collection in the same fashion as the queries were calculated. While this is reasonable up to a point, when large number of documents are added to the collection, the SVD should be recomputed to capture the new term-document correlational structure. LSI and the VSM can be directly compared in retrieval experiments when they are applied using the same term-weighting strategy.

4.3.5 Performance of LSI in Retrieval Experiments

Using a reduced number of LSI factors for retrieval clearly results in a loss of information, at least when compared to the full term representation of documents. The key question is whether the missing information is important for retrieval purposes or merely excess noise that can be eliminated without reducing performance. Deerwester et al. [19] compare LSI to the Vector Space Model on two small test collections and find that a reduced representation performs at least as well and sometimes better than the full term model.

Experiments by Dumais [20] on the TREC corpus, a very large heterogeneous document collection, obtained very good results using LSI for the routing problem, scoring slightly below the best system which used massive query expansion and the Vector Space Model. The performance of LSI for the adhoc search task was not as good, but it was still on par with the performance of the average system evaluated at the TREC-2 conference. These experiments indicate that the LSI representation has the potential to perform as well as the Vector Space Model. Considering the dramatic reduction in the dimension of the document representation, we can anticipate that LSI can be used in conjunction with statistical classification to significantly improve retrieval performance, at least in situations where enough training data is available (i.e. the routing problem).

Chapter 5

Evaluation

In the previous chapters, we have introduced several strategies for solving the information retrieval problem. With a number of reasonable choices, how can a researcher determine which strategy is most likely to be effective in practice? To judge the merits of different retrieval strategies, one must conduct some type of retrieval experiment. These experiments are usually based on a computer simulation using preconstructed queries that have been manually researched over a known text collection. This chapter will describe the standard approaches to evaluating the performance of one or more retrieval strategies.

Successful evaluation using computer experiments requires the following components:

- 1) One or more document collections with a set of queries and their relevant documents.
- 2) A measure of the quality of the search based on the document similarity scores produced by the system.
- 3) A valid statistical methodology for determining whether measured differences between retrieval methods are statistically significant.

The experimental process proceeds as follows. For each query, the retrieval strategy is applied to produce a list of documents ranked in order of their predicted relevance to

the query. The list is scored according to how it ranks the relevant documents, using one of the standard evaluation measures. Scores are computed in this fashion for each query using the different retrieval strategies, and statistical hypothesis testing determines whether there is a significant difference between the retrieval methods. Each component in this process will be described at length in the following sections.

5.1 Test Collections

A test collection that is useful for retrieval experiments contains three major components: a collection of documents, a set of queries, and an enumeration of the documents in the collection that are relevant to each query. The relevant documents are determined manually, and provide the standard against which all retrieval systems are measured. The optimal retrieval system, if it existed, would rank all documents designated as relevant higher than those designated as non-relevant for all queries. The queries are designed to simulate questions that might be asked to the system in a realistic setting or they may represent actual user queries.

5.1.1 Important Characteristics of the Test Collection

A large number of test collections have been developed over the years for use in retrieval experiments. They differ very widely in terms of size and content. The best choice of retrieval strategy will depend, at least to some extent, on the characteristics of the collection, so the goal will be to find a method that performs well in a wide variety of situations. Among the important properties that vary between test collections:

- 1) number of documents
- 2) style and content of collection
- 3) length and specificity of query
- 4) number of relevant documents

5) length of documents

The size of the collection is important for measuring the performance of a system. Computationally intensive methods may perform well for small collections but are of limited use on very large collections. Systems are often refined by adding a screening step to get rid of the documents that are most clearly not relevant. Collection size also plays a role in issues such as inverted index storage and access, document representation by Latent Semantic Indexing, and document scoring algorithms.

The distribution of words over documents depends heavily on the style and content of the collection. Some collections, such as news articles, will be loaded with proper names which could be very useful if recognized. Words that are rare in general use may be commonly used in a homogenous collection, and vice-versa. Documents may cluster based on common stylistic queues rather than similarity of content, which can make it harder to use relevance feedback on heterogenous collections. A system that is optimized for a particular collection may fail to perform as well on another collection with a different word-document distribution.

The best retrieval strategy may depend a lot on the length and specificity of the query. Complex data-driven retrieval strategies may have little success with short queries and limited amounts of information. However, the detail that they provide may be vital to obtain good results for longer, more precisely defined queries. Queries which share little vocabulary with their relevant documents may require that the system have some language understanding capability in order to discover relevant documents.

The number of relevant documents does not influence the evaluation of an adhoc query, but it is very important for routing or relevance feedback. Even the best relevance feedback strategy may not be particularly helpful if there are very few relevant documents to discover. The number of relevant documents (number of observations) is particularly important when using statistical classification for the routing problem. In general, it is better to consider a simple retrieval strategy if there is not enough evidence to support a more complex model.

Document length can be an important issue in the similarity ranking of documents. Clearly, if documents have widely different lengths, then it is important to have a

reasonable document normalization strategy. In addition, long relevant documents may be diluted with lots of unrelated information, and thus receive a much lower similarity score. Shorter documents may display higher variability in their similarity scores and thus some will tend to be ranked higher than longer documents. This issue motivates much of the work on passage-level retrieval that is discussed in a previous chapter.

5.1.2 Problems with Test Collections

There has been a lengthy debate in the IR community about how well computer experiments using a fixed set of queries over a few commonly studied test collections accurately model real user searches. Clearly, it is impossible to characterize the full range of user interests and query-construction strategies with a small and limited number of queries. It is important to examine how closely the current evaluation strategies come to this goal. Salton [68] summarizes the most significant concerns about the validity of computer experiments, given below:

- 1) The queries used for test purposes may not represent actual user search requests. The relevance assessments with respect to these test queries may not be representative of the judgements that would actually be obtained in realistic search environments.
- 2) The assessment of relevance, or utility, of a document with respect to a query must necessarily be tailored to individual users. Hence the retrieval results obtained with any given set of relevance data may be valid only for one particular set of users and not be more universally applicable.
- 3) Some evaluation measures (recall) are based on knowledge of the complete set of relevant documents with respect to each query, including both retrieved and unretrieved items. The sampling methods used to choose documents to be evaluated are suspect, so unrealistic results or measurements that favor particular search methodologies may be produced.

- 4) The query and document collections used for test purposes are often unrealistically small; in such cases it is unrealistic to assume that the laboratory results are valid for real searches performed under operational conditions.

Since the cost for manual evaluation of relevant documents is high, test collections will only contain a limited number of queries. This number will probably be far too small to cover the full range of topics included in the collection. In addition, these queries will not necessarily model the properties of search requests for other document collections. The first problem is not serious if the actual query base represents an unbiased sample of the universe of all reasonable queries for the collection. There is no good solution to the second problem other than to test retrieval methods on a wide variety of different test collections.

While the relevance of a particular document to a query is usually obvious to any individual reader, it is based on an intuitive notion and the concept of relevance is difficult to quantify [71]. Schamber et al. [72] divide it into two distinct categories, topical relevance and situational relevance. Park [59] describes the distinction as follows: “Topical relevance is context-free and is based on fixed assumptions about the relationship between a topic of a document and a search question, ignoring an individual’s particular context and state of needs.” Situational relevance incorporates a wide variety of factors, such as the user’s experience and background and the nature of their information need.

An evaluation framework based on test collections can hope, at best, to capture the notion of topical relevance. Since the goal of retrieval experiments is to find the system that maximizes performance over the entire population of users, it makes sense to ignore situational relevance. However, human evaluation of relevance of a document-query pair may often be based on a single user judgement, some degree of situational relevance will inevitably be introduced. This problem can best be addressed by having multiple readers determine the relevance of documents. Alternatively, the query can be specified very precisely, so that debate over relevance can be minimized. The topics supplied with the TREC collection fall into this category [35]. However, this style of query formulation does not necessarily reflect typical user behavior.

For large text collections, it is impossible to manually evaluate the relevance of every document in the collection with respect to each query. Generally, some screening process will be used to select the documents suitable for examination, and all unexamined documents are assumed to be not relevant. Therefore, it is inevitable that some relevant documents will slip through the net. This means that any measure based on recall, which looks at the total number of relevant documents, will not be accurate. In particular, the true number of relevant documents will be underestimated. This alone does not cause too much trouble for a retrieval experiment, since all methods will be evaluated based on the same set of relevant documents. However, depending on how the documents were selected for examination, there may be a bias for or against particular retrieval strategies.

Most of the test collections used in the early years of retrieval experiments were extremely small, consisting of at most a few thousand documents. This was due primarily to limited computing power available in that era. It is possible that some of these results will not generalize to the larger collections in common use today. However, the important question once again is whether any bias is introduced in the smaller test collections. Clearly there are some problems. The early test collections consisted primarily of journal abstracts or other short summaries, which have been demonstrated to have different properties than long full-text documents. However, the recent development of the TREC document collection [35], which contains two gigabytes of documents of widely varying lengths, has helped to mitigate this problem.

5.1.3 The Alternative - User Studies

Given the problems associated with experimental test collections, it is fair to ask whether there are any reasonable alternatives. The best alternative is to replace the fixed queries with actual users, and measure their satisfaction with the output of the retrieval system. User studies will more accurately simulate the retrieval system in operation and may give a better reflection of system performance. For example, it may take only take a single document to satisfy the user's need while additional documents, while technically relevant, do not provide any more useful information.

However, user studies have a number of drawbacks when compared to computer simulations. They require considerably more effort to implement, since a significant number of subjects must be found, and their response to search results must be recorded in considerable detail. It is difficult to compare results between users, since each subject will have their own standard of what constitutes a successful search, and it is difficult to compare results between studies, since each study will have a different set of users. In order to directly compare two retrieval strategies, one must present the results of both to a single subject, who must then decide which is better. This is a much more difficult and subjective task than a computer experiment. Finally, in order to make a precise comparison, one must quantify the user's opinion of the quality of the search, which adds another level of subjectivity (variability) to the evaluation process.

The user's opinion of a system will be based on a large number of factors, not just the performance of the underlying retrieval engine. The quality of the user interface and the ease of use of the system for developing and modifying the query will also be important factors. While these components are clearly important to the overall success of the retrieval system, they will make it more difficult to compare between document ranking strategies. Studies have found that user satisfaction is not highly correlated with some of the traditional measures of retrieval performance [76], indicating that simulation studies using test collections may not accurately measure the effectiveness of the retrieval system.

Hersh [38] suggests that there are more useful approaches to system evaluation than simply measuring the number of relevant documents retrieved by the search, since this may not correspond to success or user satisfaction. An alternative approach is to ask subjects to use the retrieval system to find the answers to a series of questions, given in the form of a test. If the subject can successfully answer the test questions by consulting the system, that is a good indication that the system is working. An experiment based on question answering adds a strong quantitative component that is missing from other forms of user studies. It still measures overall system performance and does not distinguish between ease of use and the quality of the retrieval algorithm. The researcher must also account for inter-subject differences.

The experiments in this thesis are designed to find new retrieval strategies to accurately identify relevant documents. Since other components of the retrieval system, such as the user interface, have no significance in this context, it is important that we evaluate solely on the basis of the retrieval algorithm. Therefore, we are best served by using computer experiments on test collections. The remainder of this chapter will be devoted to describing the evaluation measures and testing necessary to accurately measure the performance of the retrieval system based on a computer experiment.

5.2 Performance Measures

The standard document retrieval strategy will process the query and return a list of documents to the user, ranked in order of their similarity to the query. For each query in the test collection, these documents are identified as either relevant or non-relevant. In order to assess the performance of a system, we want to assign a score to each of these rankings based on the degree to which relevant documents are highly placed in the ranking.

For most test collections, there is only a binary notion of relevance. Each document is identified simply as relevant or non-relevant, and there is no sense of an amount of relevance. While additional information about the amount of relevance might be helpful, it would add an additional layer of subjectivity to the evaluation process. The value of a particular relevant document may depend on the information already available to the user. For most evaluation methods, one can assume the relevant documents are indistinguishable from one another, and the same can be assumed for non-relevant documents.

5.2.1 Precision and Recall

The simplest approach is to assume that user wants to examine a fixed number of documents, which we will call the document cut-off value (DCV). Given the DCV, then the goal is maximize the number of relevant documents in the retrieved subset. Clearly, any reasonable evaluation measure should be monotonic in the number of

relevant documents retrieved. The most commonly used measures are known as precision and recall and are defined below:

	Relevant	Non-relevant	
retrieved	r	$n-r$	n
not retrieved	$R-r$	$N-n-R+r$	$N-n$
	R	$N-R$	N

$$\text{Precision} = \frac{r}{n} \quad \text{Recall} = \frac{r}{R}$$

Precision measures the quality or efficiency of the search, in terms of the fraction of useful material in the set of retrieved documents. Recall measures the breadth of the search, expressed as the fraction of the available relevant material presented to the user. It is generally assumed that document scores have a strong linear ordering, which means that no two (or more) documents receive the same score. Computing precision or recall at a DCV which forces the system to choose from a group of documents with identical scores is a much more complicated process [62].

Precision and recall are both directly proportional to the number of relevant documents retrieved; they differ in terms of the proportional weight assigned to each additional relevant document. Both measures are normalized to range between zero and one, which is necessary so that the measures will have a consistent scale over different queries with different numbers of relevant documents. With this normalization, the concept of average precision and average recall over the queries in a collection becomes a meaningful concept.

Precision and recall have an inverse relationship to one another. As the number of documents examined increases, recall must increase, but in general precision will tend to decrease, following the average likelihood of relevance of each additional document. The most common display of retrieval performance is known as the precision-recall curve, and is simply a scatter-plot of precision vs. recall, evaluated at each point in the ranking where a new relevant document is discovered. Alternatively, one could plot precision or recall vs. the document cut-off value.

5.2.2 Averaging Methods

Each query generates a separate evaluation curve. The overall performance of a system is judged by averaging these curves. For plots of precision or recall versus the DCV, this can simply be done by averaging at the same DCV for each query. Averaging curves which plot precision vs. recall is a bit more complicated, since the defined recall levels (which depend on the number of relevant documents) are different for each query.

The most common averaging strategy, known as macro-evaluation [79], is to measure precision at the same fixed levels of recall for each query and build the average curve by taking the average precision over all queries measured at each of these recall level. In order to make sure that the fixed levels of recall are defined for each query, it is necessary to perform some interpolation of the individual precision-recall curves.

Assume (R_i, P_i) is a recall-precision pair evaluated at a DCV equal to the i 'th ranked relevant document. Then the traditional interpolation strategy is to define [79]:

$$P(R^*) = \sup P : R \geq R^* \text{ and } R \text{ in } \{(R_i, P_i)\}$$

In other words, define the precision at the interpolated recall level as the largest value of precision for any pair with recall greater than the given level. Once these interpolated values are obtained for each query, they can simply be averaged to produce the overall precision-recall curve.

5.3 Assumptions in Evaluation

The precision-recall (P-R) curve defines a bivariate measure of retrieval effectiveness. How do we determine when one method is better than another on the basis of this measure? A standard technique for comparing the scores of a bivariate measure is to fix one variable and judge performance with respect to the other variable. The fixed variable is often called the control or independent variable while the other is labelled the response or dependent variable. The two approaches to evaluation that are being compared have different definitions for these variables. The average P-R

curve produced by macro-evaluation defines recall as the control variable and precision as the response variable, since precision is averaged at a fixed level of recall. Averaging at a fixed DCV uses the number of documents examined as the control variable and precision or recall as the response variable. This results in substantially different interpretations of the difference between evaluation scores when they are averaged over a number of queries.

5.3.1 Normalization

The P-R curve uses recall as its measure of equivalence between queries. By judging methods on the basis of their average precision, the researcher is assuming that a particular recall level must be attained by every query, and that the best search method is the one that attains this recall level most efficiently, i.e. with the fewest number of non-relevant documents. In some contexts, such as a patent search, these are probably reasonable assumptions, but for other applications, recall is not necessarily a good measure of equivalence. Consider an example where one query has 20 relevant documents while another has 200. For most users, 50% recall would be a very reasonable goal in the first search, but it is likely to be totally unmanageable in the second. Therefore, one might question the validity of an average P-R curve when similar recall levels are based on very different numbers of relevant documents.

The DCV-based methods normalize on the basis of equivalent effort, based loosely on the number of documents examined, rather than equivalent performance. One can imagine that a searcher might be willing to examine only a fixed number of documents for any particular query. This approach also makes it possible to measure differences between search methods in terms of precision or recall. When a DCV-based method is averaged over multiple queries, using average precision weights each relevant document equally in the final score while using average recall assumes that relevant documents from queries with a larger relevant set are less important than those with a smaller relevant set. Users can select which measure best suits their search philosophy.

Clearly, neither method of evaluation dominates the other. Actual user behavior

is likely to be based on some combination of the two methods. The user will probably have a limited tolerance for examining new documents (measured by the DCV), but this tolerance may be greater or smaller depending how much relevant information has already been obtained (measured by recall). Note that the P-R curve provides no information about the average number of documents that must be retrieved in order to obtain a given level of recall. DCV-based methods have their own advantages and disadvantages, as will be explained in subsequent sections. The author believes that averaging on the basis of equivalent effort, rather than equivalent performance, is much more reasonable for most retrieval applications.

5.3.2 Problems with the DCV

Both precision and recall are normalized to range between zero and one. This means that they can be directly compared between queries and between systems. However, when precision or recall are calculated using a DCV, they are often restricted to a small fraction of that range. This is a problem for precision with large document cut-offs and for recall with small document cut-offs.

If the DCV is larger than the number of relevant documents, it is impossible for the precision to be one. In particular, if all relevant documents have been retrieved, the precision will continue to decline with an increasing DCV, even though the search algorithm is doing the best it can. Comparison of average precision between different queries and different search methods can then be misleading. For example, one method may retrieve all relevant documents in the top ten while another may not achieve full recall until 30 documents have been examined. If the DCV is 30, these two methods will have the same average precision. One could argue that if the user was planning to examine 30 documents anyway, then it doesn't matter which method finds them earlier. However, this does suggest that users should make quality judgments on the basis of more than one level of the DCV, just as precision is measured at many levels of recall for the P-R curve.

If the DCV is smaller than the number of relevant documents, it is impossible to obtain a recall value of one. For example, if only five documents are examined

and 50 relevant documents exist for the query, then the recall is only 10% even if all documents examined are relevant. This can make search methods appear much worse than they actually are. For example, another search method might return only two relevant documents in the same search described above. A comparison of 4% and 10% recall might suggest very little difference between methods, although with a higher DCV, the first method might turn out to perform much better.

One way to address this problem is to measure precision or recall over a range of DCV's and then average the results [40]. It is relatively simple to compute the statistic at all integer values of the DCV in a range of interest and then use the average over this range as the final measure of performance. For example, rather than just measuring precision at 10 documents, one might compute precision averaged over all values in the range [1-20]. This measure will smooth out any irregular effects due to the choice of exactly 10 rather than say 8 or 12 as the DCV of interest.

The relationships shown above between precision, recall, DCV, and the number of relevant documents suggest that precision is a more accurate measure for low DCV and that recall is more accurate for high DCV, where the definition of low and high is relative to the number of relevant documents. User preferences would be likely to follow this pattern. Searchers choosing a low DCV would be more interested in precision while those choosing a high DCV would be more interested in recall. Therefore, as long as the researcher chooses between precision and recall based on the size of the DCV, and then averages this measure at a range of different levels, there should be very little loss of power from using this method. Note that the P-R curve does not suffer from this problem at all.

5.3.3 Relevance Feedback and the P-R curve

In relevance feedback, a revised query is generated by updating term weights and/or adding terms from relevant documents found in a previous search. If the feedback search is re-evaluated under the same conditions as the original search, feedback will almost invariably improve performance at low recall as measured by the P-R curve. The similarity scores of any relevant documents that are used to generate the query

will increase, which will improve the ranking of these documents relative to the others in the corpus. This phenomenon, known as the ranking effect, does not improve the quality of the search from the user's perspective, since it merely reorders documents that have already been examined.

There are a number of different methods for revising the P-R curve in order to estimate the actual improvement due to relevance feedback. The SMART [66] system uses rank freezing, which means that all documents that have been examined by the user are frozen at their original ranks and all new documents are given ranks below them. Several alternatives to rank freezing are also proposed in the SMART study. Modified freezing only fixes the ranks up to the last relevant document that is used for feedback. The residual collection method removes all previously found relevant documents from the collection and computes precision and recall as if they never existed. The test and control method divides the corpus into two roughly equivalent samples. The first is used for relevance feedback, while the second provides untested data for evaluation.

Several of the methods have clear drawbacks. Residual collection may perform poorly when the number of relevant documents is small, since the removal of documents from the relevant set will have a significant impact on the evaluation results. The test and control method is only unbiased if the two samples have equivalent numbers and distributions of relevant documents. Finding a partition of the collection that would be acceptable for all queries is a very difficult task. Otherwise, there does not seem to be a common standard for evaluating relevance feedback with respect to the P-R curve.

If this problem is considered from the user's perspective, the solution is clear. Once a user has expended the effort to examine a document, whether relevant or non-relevant, that document should be counted for the purpose of evaluation. Rank freezing performs exactly this function, since documents are ranked in the order that they are examined by the user.

This approach is easy to adopt for evaluation based on the DCV. Different feedback methods are compared when they have examined the same number of documents. For example, assume the DCV is fixed at 20. One method might retrieve the 20

highest ranking documents and compute precision or recall. To incorporate relevance feedback, retrieve 10 documents in the first step, revise the query and then find the 10 highest-scoring documents that have not yet been examined. The combined set of 20 documents from the feedback run can be directly compared to the initial method without introducing any source of bias. The two methods required the same amount of effort from the user, and thus are directly comparable.

5.4 Univariate Measures of Performance

While the average P-R or DCV curve describes the performance of a system reasonably effectively, it does not lend itself easily to statistical hypothesis testing. It is difficult to measure what defines a significant difference between two performance curves. Therefore, it is common to summarize the full performance curve with a single evaluation score. This score can be computed for each query and methods can be compared on the basis of how consistently one scores better than another over the full set of queries.

5.4.1 P-R based

The most common technique is simply to compute the average precision at multiple levels of recall. For example, Salton and Buckley [69] use average precision at 25%, 50%, and 75% recall. The TREC-2 [35] evaluations average the precision scores at every multiple of 10% recall. While using a one-dimensional measure obscures important information, such as whether there is any interaction between performance and recall, it does make it easier to compare a large number of different retrieval strategies. One can summarize a DCV-based evaluation curve in a similar fashion by averaging precision or recall over multiple cut-off values.

Alternatively, if interested in performance at a specified level of the control variable, one could just select that point on the curve and use the value of the response variable. Other summary statistics based on precision and recall have also been proposed. Some evaluation studies have selected the point on the curve where precision

equals recall. This method captures improvements in both precision and recall.

Van Rijsbergen [79] developed a measure that allows the user to weight the relative importance of precision and recall. One such formulation of this measure is given below:

$$F = \frac{P * R}{\alpha * R + (1 - \alpha) * P}$$

α ranges between zero and one and measures the relative importance of precision versus recall. This measure is generally computed at a fixed cut-off value. The full evaluation curve is often summarized by selecting the maximum value of F at a number of different cut-off values.

5.4.2 Other Measures

A number of other evaluation measures have been proposed that are not directly related to precision and recall. Swets [77] suggests that documents scores can be modelled by two normal distributions, one each for the relevant and the non-relevant documents. The quality of a retrieval strategy is determined by its ability to separate the mean of the relevant and the non-relevant documents. This can be measured by:

$$S = \frac{\bar{x}_{rel} - \bar{x}_{nrel}}{\sqrt{s_{rel}^2 + s_{nrel}^2}}$$

S is similar to a two sample t-statistic, however the sample variances are not scaled to reflect the sizes of the populations. It is unclear why Swets decided not to scale the variances.

While the two-sample t-statistic looks like a plausible measure, it has a number of drawbacks. Non-relevant document scores will almost always be skewed, with a heavy tail in the direction of higher scores. All of the important action takes place in the tail of the non-relevant score distribution, so the mean is probably not the best measure of this behavior.

The measures based on precision and recall operate under the assumption that the user's need can be expressed in the form:

- (1) I want to find x% of the relevant documents as quickly as possible (P-R curve).

- (2) I want to find as many relevant documents as possible, but I'm only willing to examine n documents (DCV curve).

A better expression of the user's goals might be:

- (3) I want to find one relevant document or . . .
- (4) I want to find k relevant documents

Cooper [14] introduces the concept of Expected Search Length (ESL) which measures the expected number of documents that must be examined to satisfy the user's need. If the documents are strongly ordered (no ties), the search length can be calculated directly. If there are ties, one must then estimate an expected value. The retrieval strategy with the smallest ESL can be considered optimal for a request specified like (3) or (4) above. The only problem with this method is that there is no concept of recall, since the total number of relevant documents plays no role in the computation. Some would not consider this to be a problem, since recall is very difficult to estimate for real user searches.

Precision and recall are based on a binary assumption of document relevance. While this is the standard representation used in test collections, it may not capture the concept of relevance from the perspective of the user. A more intuitive approach is to ask the user to provide a list of preferences among the examined documents. A preference statement merely states that document A is better than document B. If enough preference relations are accumulated, this may provide a quite accurate picture of the actual importance ranking of the relevant documents.

Frei and Schauble [24] suggest modelling the performance of a retrieval system based on the number of the user's preferences which are satisfied. From the ranked list of documents produced by each retrieval strategy, one can compute the number of preference relationships that are satisfied. The strategy that satisfies the most preference relations is considered to be best for that particular query. One can apply this measure to test collections by assuming that each relevant document is preferred over any non-relevant document.

Clearly, one can choose from a wide variety of different evaluation measures. No one measure dominates all the others. In general, there is a high degree of correlation

between the different measures. The best choice depends on a wide variety of factors, including the user's conception of the goals of the search and the relevance information available. Another factor to consider is the interpretability of the results. For many of the measures described above, it is difficult to determine the meaning of an observed difference between retrieval strategies. For many users, this may be the single most important factor in determining the value of an evaluation measure.

5.5 An Alternative Method of Evaluation

There are a significant number of researchers (including the author) who are not satisfied with any of the current evaluation measures. Most of the author's concern is related to the assumptions that go into the representation of documents as either relevant or not relevant. This representation leads to two primary assumptions:

- (1) All relevant documents are equally important.
- (2) The information contained in relevant documents is additive, i.e. two relevant documents contain twice as much useful information as one relevant document.

In most cases, these assumptions will not be strictly satisfied, and the question is whether they provide a reasonable approximation to the actual distribution of relevant information. It is likely that some documents will be more valuable than others, but without any additional information, there is no better alternative than assigning an equal weight to each relevant document. The second assumption is much more difficult to justify.

A priori, there is no reason to assume that information in relevant documents will be additive, since it is very likely that some of documents will overlap, i.e. have some material in common. The combined worth of two documents will almost always be less than the sum of their values. This means that measures based on the number of relevant documents retrieved will be biased significantly in favor of strategies that find more relevant documents, and will make these methods look much better than they actually are.

It is possible that two documents could be more valuable together than separately, if for example, reading both of them allows you to make some interesting connection that would not be possible when examining one without seeing the other. However, these occurrences are likely to be extremely rare when compared to the number of times two documents contain information which overlaps.

This problem can be addressed by looking at the issue from a set-theoretic perspective. Define a space S which describes all the information in the corpus. For any particular query there exists a subspace $R \subseteq S$, which defines the contents of the corpus that are relevant to the query. Each document d_i has a corresponding information space D_i which is also a subset of S . We can also define a value measure V , which measures the value of any information set with respect to the query, and define $V(R) = 1$ and $V(R^c) = 0$. A document can be considered relevant if R and D_i overlap, i.e. if $V(D_i) > 0$. In this context, it is possible that $V(D_i) = 1$, indicating that one relevant document contains all the relevant information in the corpus and that all other relevant documents do not provide any additional relevant information. The union of all relevant documents will contain R by definition, but some relevant documents may be expendable without reducing the information content. In particular, it is likely that the information space of many relevant documents will overlap.

Using this model, we can produce new definitions of precision and recall:

$$\text{Recall}(D_1 \dots D_n) = V(\cup D_i) / V(R)$$

$$\text{Precision}(D_1 \dots D_n) = \{\min k \mid V(\cup_k D_i) = V(\cup_n D_i)\} / n$$

In other words, recall measures the value of the documents retrieved compared to the value of all of the relevant documents, while precision compares the number of documents retrieved to the minimum number of documents that must be retrieved to find the same relevant information. These two measures are likely to be a much more accurate representations of the quality and efficiency of a search than a binary definition of relevance.

What are the implications of this approach for conventional retrieval experiments? It is not intended to imply that current experiments are useless, merely that they

will overestimate the importance of obtaining large numbers of relevant documents in many contexts. A simple search may well be completely satisfied with a single relevant document. However, there are some applications where relevant documents can be considered independent and additive in nature. It may be important in patent research, for example, to find all instances of prior art. The TREC queries [35] generally request all documents that satisfy a particular set of criteria.

The real problem with using this model is that queries and documents must be analyzed in far more detail to produce evaluation results. The information in all relevant documents must be pooled, and each component assigned a weight indicating its degree of importance. The experimenter then identifies which components appear in each relevant document. This type of analysis introduces a whole new level of subjectivity into the retrieval process. If one is already unhappy with the subjective nature of relevance judgements, then a more complex evaluation system is only going to increase this variability.

A more accurate description of the relevant information in a document collection might well lead to better evaluation of retrieval systems, but the cost of producing queries suitable for this style of evaluation is high. The knowledge that is gained by the in-depth analysis of a few queries might be outweighed by the coverage of the collection that is lost by being forced to work with a much smaller set of queries. Researchers should simply be aware that current evaluation methods are heavily biased towards quantity rather than quality.

5.6 Hypothesis Testing

Once a number of different retrieval strategies have been applied to a test collection and an evaluation score computed for each query, then the researcher may want to know whether the results indicate that one method is significantly better than another. Hypothesis tests can be used to determine whether the observed differences are really meaningful or are simply due to chance. For most scoring methods, it is impossible to make this distinction for an individual query. However, when systems are evaluated over multiple queries, it becomes possible to measure the variability

associated with each evaluation measure and determine whether one method performs consistently better than another.

5.6.1 A Brief Introduction

It is useful to briefly define the concept and terminology of hypothesis testing [47]. The goal is to determine whether we can reject a default assumption about the data. There are only two alternatives, accepting or rejecting the hypothesis. Let Ω define the set of all possible alternatives and divide Ω into two mutually exclusive subsets Ω_0 and Ω_a . Our null hypothesis H_0 for an unknown variable θ is that $\{\theta \in \Omega_0\}$.

There are two possible errors that can be made: rejecting the null hypothesis when it is true, or accepting the null hypothesis when it is false. A test cannot minimize both types of error at once. Generally, hypothesis tests are designed so that the probability of falsely rejecting H_0 will not exceed a value α , called the significance level of the test. The probability of falsely accepting H_0 is called β , and the power of the test is defined as $1 - \beta$. The power measures the ability of the test to detect a true difference.

A statistic X which measures θ is computed from the data and X is compared to a reference distribution which describes θ under the assumptions of the null hypothesis. It is then possible to calculate the probability that a value as extreme as X could occur in the distribution of θ , given H_0 . If it is less than α then H_0 is rejected, indicating that it is extremely unlikely that the assumptions made about θ were accurate.

In our context, the null hypothesis will be that there is no difference in performance between retrieval strategies and we use a significance test to determine whether we can reject this hypothesis. Significance tests can be divided into two major categories, parametric and non-parametric. Parametric tests assume that we can model the scores using a common probability distribution. Non-parametric tests do not rely on an underlying model and are generally based on ranks that are derived from the data. Parametric tests tend to be more powerful because a lot of information is lost when only ranks are used to test the data, but they can run into problems if the model assumptions of the test are violated.

Rather than defining an explicit value for α and describing the test results as either positive or negative, we will instead simply compute the p-value, or probability that the measured value could have occurred given H_0 , and use it to describe our degree of confidence in the null hypothesis. A very small p-value will indicate that there is extremely likely to be a difference between methods. The significance tests will be divided into two categories, comparisons between two methods and comparisons between more than two methods.

5.6.2 Comparisons between two methods

In this example, the experimenter wishes to compare between two retrieval methods and decide if one is significantly better than the other. The measured values for both methods will depend on the properties of the query, therefore they should not be treated as independent samples from a single underlying distribution. Instead, the measurements should be viewed as matched-pairs of observations for each query. For comparison purposes, the values of the evaluation measures are not important, so attention will be restricted to the difference between measures:

$$D_i = Y_i - X_i$$

where X_i and Y_i are the scores of retrieval methods X and Y for query i . Three different significance test will be considered, the t-test as well as its nonparametric alternatives, the Wilcoxon signed ranks test and the Sign test.

5.6.3 The Significance Tests

The standard approach in this situation is to apply the paired t-test to the differences. This test requires the following assumptions:

- D_i 's are mutually independent
- D_i 's are normally distributed

The test statistic is:

$$t = \frac{\bar{D}}{s(D_i)/\sqrt{n}}$$

which under the hypothesis $\mu(D_i) = 0$, follows the t-distribution ¹.

The most simple non-parametric test for matched pairs is the sign test. It uses only the sign of D_i , ignoring information about the magnitude of the difference. The assumptions of the test are:

- D_i 's are mutually independent
- The measurement scale is at least ordinal within each pair, i.e. the relations '>', '<', or '=' can be determined

Define the statistics s^+ and s^- as the number of queries with $D_i > 0$ and $D_i < 0$ respectively, and let $n = s^+ + s^-$. Any cases where $D_i = 0$ are ignored. Under the hypothesis that:

$$P(X_i > Y_i) = P(X_i < Y_i) = \frac{1}{2}$$

s^+ is distributed as Binomial($n, 1/2$). Significance tests can be conducted using a Normal approximation. The final test statistic becomes:

$$T = \frac{2s^+ - n}{\sqrt{n}}$$

which is compared to the standard Normal distribution.

Rather than ignoring the magnitude of D_i 's, as is done in the sign test, the Wilcoxon signed ranks test uses the ranks of the D_i to test for significance. The assumptions are:

- D_i 's are mutually independent
- The measurement scale of the D_i 's is at least interval
- D_i 's are symmetric and all have the same median

¹

$$\bar{D} = \frac{1}{n} \sum_{i=1}^n D_i, \quad s(D_i) = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (D_i - \bar{D})^2}$$

Define the statistic:

$R_i = \text{sign}(D_i) * \text{rank}|D_i|$ and compute,

$$T = \frac{\sum R_i}{\sqrt{\sum R_i^2}}$$

which for a sample size of n and no ties, can be simplified to:

$$T = \frac{\sum R_i}{\sqrt{n(n+1)(2n+1)/6}}$$

Under the hypothesis, $\text{median}(D_i) = 0$, this statistic will have the standard Normal distribution.

5.6.4 Optimal two-sample tests in IR

The key question is which of these tests is optimal for information retrieval applications. The t-test is based on the assumption that the differences are described by the normal distribution. Strictly speaking, this cannot be true for our standard measures of precision and recall, since they are both discrete and bounded. However, it is well known that other discrete bounded distributions such as the binomial can be well approximated by the normal distribution. If the measure in question is the average of many discrete values, such as precision averaged at nine levels of recall, then it should be almost indistinguishable from a continuous variable.

Furthermore, the t-test is relatively robust to many violations of normality. Potential problems, such as skewness (lack of symmetry) or outliers (very large or very small values), can be tested by making a quantile plot of the data. A quantile plot compares the ordered data to the expected value of a sample of equal size from the standard normal distribution. If the normal distribution is reasonable, the points should fall on a line.

Unfortunately, in the author's experience, outliers tend to be a common problem in retrieval experiments. Many queries give poor results regardless of which method is applied, and the observed improvements in average retrieval performance often boil down to a large difference for one or two queries. While these differences may be

real and important, it is hard to make a general statement about the superiority of a particular method on the basis of such results. This problem suggests that non-parametric tests may be a more desirable alternative.

When choosing between the Wilcoxon and the sign test, it is interesting to examine their asymptotic relative efficiency (ARE) compared to the t-test. The ARE measures the limit in the sample size ratio needed to make the two tests equivalent under an equal error-rate assumption. The ARE of the Wilcoxon test never drops below .864 with respect to the t-test, while it drops to 1/3 for the sign test for some distributions [13]. Since the sign test relies only on the sign of the difference between methods, it should not be surprising that it is less powerful than the alternatives. This indicates that the Wilcoxon test is much more robust than the sign test in most experimental situations, and we will rely primarily upon it in our two-sample retrieval experiments. From a practical standpoint, all of these tests are easy to compute, so there is no reason not to try all of them and look more closely at the data if there is significant disagreement among them.

5.6.5 Comparisons between more than two methods

The situation changes when there are more than two retrieval methods to be compared. A standard approach to this problem is to examine the data using a two-way Analysis of Variance (ANOVA), which simultaneously tests for differences in the average score of each method, correcting for the effect of the queries. What advantage does this have over performing a two-sample test for each pair of methods? First, the number of pairs grows very rapidly with the number of methods, so performing only one test saves a lot of work. Second, there is a problem with multiple comparisons. The more paired tests that are conducted at the same significance level, the greater the chance that at least one of the paired tests will produce a significant result when there is actually no difference between the methods.

There are several non-parametric variations of the ANOVA. In general, these methods replace the data with ranks based on the relative performance within each query. While using ranks reduces the power of the test, it requires less restrictive

assumptions about the error distribution. Only one nonparametric test, the Friedman test, will be described here. Others can be found in Conover [13] or Hollander and Wolfe [39].

All of these tests assume that the query effect and the effect of the evaluation methods are independent and additive. An initial test is performed to determine whether there is any difference between methods. If a difference is detected, a subsequent multiple comparison test determines which methods are significantly different.

The initial test statistic for the testing methods has the form: $F = \text{MSR}/\text{MSE}$, where MSR is the mean-squared difference between retrieval methods and MSE is the mean-squared error. If the average difference between retrieval methods is large relative to the average error, then one can assume that the retrieval methods are significantly different. The value of this statistic can be compared to the F-distribution, which is the distribution of these scores if all retrieval methods are equally effective.

Assume the data consists of observations y_{ij} , which correspond to the retrieval performance of method j for query i , where $i = 1 \dots n$ and $j = 1 \dots m$. Let $\bar{y}_j$ represent the average score for method j and $\bar{y}$ equal the overall average. Let R_{ij} represent the rank of method j with respect to the other methods in query i and define $R_j = \sum_{i=1}^n R_{ij}$.

The standard model is:

$$y_{ij} = \mu + \alpha_i + \beta_j + \varepsilon_{ij}$$

This means the each observation y_{ij} can be broken down into: the true mean performance μ , the query effect α_i , the retrieval method effect β_j , and the error ε_{ij} . The query effect and method effect are assume to be independent and additive.

Two-way ANOVA

$$F_A = \frac{\text{MSR}}{\text{MSE}} = \frac{\frac{n \sum_j (\bar{y}_j - \bar{y})^2}{(m-1)}}{\frac{\sum_{i,j} (y_{ij} - \bar{y}_i - \bar{y}_j + \bar{y})^2}{(n-1)(m-1)}}$$

tests for $\beta_j = 0$ for all j .

assumption: errors are independent and normally distributed

distribution under H_0 : F distribution with $m - 1$ and $(n - 1)(m - 1)$ degrees

of freedom

If the F-test is significant, use multiple comparisons below.

ANOVA multiple comparisons

$$|\bar{y}_k - \bar{y}_l| \sim q_{m,\nu}^\alpha s / \sqrt{n}$$

Tukey's studentized range test

distribution under H_0 : $q_{m,\nu}^\alpha$ the studentized range distribution with $\nu = (n - 1)(m - 1)$ at significance level α and $s = \sqrt{\text{MSE}}$. All mean differences that are greater than the value shown are assumed to be significant.

Friedman test

$$A = \sum_{i=1}^n \sum_{j=1}^m R_{ij}^2 \text{ and } B = \frac{1}{n} \sum_{j=1}^m R_j^2$$

If there are no ties, A becomes:

$$A = nm(m + 1)(2m + 1)/6$$

The final test statistic that will be used is:

$$F_N = \frac{\text{MSR}}{\text{MSE}} = \frac{(n - 1)[B - nm(m + 1)^2/4]}{A - B}$$

assumption: errors are independent and come from the same continuous distribution

distribution under H_0 : F distribution with $m - 1$ and $(n - 1)(m - 1)$ degrees of freedom

If the F-test is significant, use multiple comparisons below.

Friedman multiple comparisons

$$|R_k - R_l| \sim t_{1-\alpha/2} \left[\frac{2n(A - B)}{(n - 1)(m - 1)} \right]^{\frac{1}{2}}$$

distribution under H_0 : Student's t distribution with $(n - 1)(m - 1)$ degrees of freedom.

A longer description of the ANOVA model, as well as tables of the F-distribution, the t-distribution, and the studentized range statistic can be found in Neter et al. [57]. Note that the test statistic F_N is not the original statistic proposed by Friedman. The original statistic was found to be inferior to F_N by Iman and Davenport [41]. More information on Friedman's test can be found in Conover [13].

Which of these tests is most applicable in the context of information retrieval? The two-way ANOVA would be preferred if its assumptions are satisfied. This can be tested by fitting the ANOVA model and examining the errors using quantile plots, just as was suggested for the two-sample case. In addition, one could look at the variability of the errors with respect to each method or with respect to each query. If the ANOVA model is correct, the error variability should be similar when compared between methods. The error variability of the queries should also share this property. In practice, it is likely that some queries will be much more variable than others, and it follows that these queries will have more influence on the performance in the ANOVA model. The researcher must decide if this is appropriate. Unequal variances will not invalidate the model because the design is balanced, meaning that there are no missing observations.

Both methods assume that the errors are continuous. As stated previously, if the evaluation method is the average of a large enough number of discrete measurements, it should be basically equivalent to a continuous variable. One easy way to check this is to look at the number of ties in the data. A large number of ties indicates that the errors are not continuous. For this reason, measures such as average precision at one level of recall, or average recall at one level of the DCV are not recommended for these testing procedures. These measures have a small number of discrete values, which may be poorly approximated by a continuous variable.

The Friedman test has an interesting property that may be considered an advantage. Since its test statistic depends only on the ranking of scores with respect to each query, the Friedman test will give the same answer for any monotonically increasing function of the number of relevant documents. Almost all evaluation measures fall into this category, including average precision, average recall and the E-measure. This invariance property provides an element of consistency that is independent of

the choice of evaluation measure. Normalization for differences between queries is accomplished automatically by using ranks.

In actual IR experiments, we find that outliers are once again a problem. Once again, for many queries performance seems remarkably insensitive to the choice of retrieval method, with differences being dominated by one or two variable queries. Therefore, we will tend to rely on Friedman's test to determine if there is significant difference between methods.

5.6.6 General Comments

It is important to make a distinction between statistical and practical significance. Rejecting the hypothesis that two measures are equivalent based on a quantitative test, such as the ones given above, results in statistical significance. Practical significance is much more difficult to measure. It can be described as a difference in performance that might be noticeable to the user. While one type of significance will generally imply the other, they can sometimes be different. A 1% improvement in recall may be detected by a powerful statistical test, but this will make little difference for the user. Conversely, a 10% difference in precision may not be statistically significant, due to an insufficient number of queries or the inherent variability of the data. Researchers should simply be cautioned to consider both the statistical significance and the magnitude of the difference in judging the effectiveness of evaluation measures.

What if the test is not significant? This does not necessarily mean that there is no difference between the methods, merely that the test was unable to detect one. This is where tests on a number of different corpora can be useful. It is worth recognizing that differences which are not statistically significant can still be important if they occur repeatedly in many different contexts [44].

Statistical tests merely measure the significance of differences in *average* performance. The average may or may not be a good summary statistic over a wide range of queries, and so it is no substitute for directly examining evaluation results on a

query by query basis. For very large experiments, this may not be feasible. Diagnostic plots are also useful for determining if any of the queries exhibit unusual behavior that deserves further study.

Chapter 6

Classification Methods

Statistical classification methods are designed to distinguish between groups (relevant and non-relevant) based on the distribution of the elements (documents) within each group. The general statistical classification problem can be described as follows. A population consists of two or more groups, and there exists a training sample for which the class of each element is known and a test sample for which the class is unknown. The goal is to produce a classification rule which will predict the class of the unknown elements. The classification rule is generated from the training sample based on a number of predictor variables that have been measured for both the known and unknown observations. The routing problem is merely a special case of the classification problem, since there are only two groups, relevant and non-relevant.

The challenge comes in producing a good set of predictor variables. If there are too many predictor variables, the classifier will overfit the training data, which will lead to poor performance on the test data. To give an extreme example, consider using terms as predictor variables. It is likely that every relevant document in the training sample has one or more unique terms associated with it. If one uses these terms to define the classification rule, then the classifier will perfectly identify the relevant documents in the training sample, but it will have no power to discriminate over the test sample.

6.1 Properties of the IR Problem

As demonstrated in chapter 4, terms do not make an ideal set of predictor variables. The most valuable terms for retrieval purposes occur in relatively few documents, so large numbers of them are needed in order to adequately characterize a document collection. The more common terms have far less power to discriminate among topics. In addition, there are thousands of terms to choose from. Variable selection can be used to reduce the size of the predictor set, but it does not address the fundamental distributional problems associated with term-based variables.

As an alternative, we will turn to the technique of Latent Semantic Indexing, which uses the singular value decomposition to produce a small set of indexing variables that explains much of the correlation in the original term by document matrix. These factors are orthogonal, eliminating problems with term dependence, and dense, capturing the relationships between a large number of terms in a single variable. It is hoped that the LSI representation, when combined with statistical classification should provide a more effective solution to the routing problem.

A great many different techniques for classification have been developed in statistics, pattern recognition, and machine learning. The first task is to identify which methods may be best suited to the IR problem. Despite the fact that LSI produces a considerable dimension reduction when compared to the original term space, the predictor space still consists of 100-200 variables. The classification method must also be able to deal with a training sample of several thousand documents. Therefore, the technique must be relatively efficient from a computational standpoint and work on large, high dimensional problems.

In order to be competitive with the Vector Space Model, the classification rule should be able to rank documents in order of their estimated likelihood of relevance. Since so many documents are available, users must be able to select the number of documents they wish to examine. In addition, a ranked output allows them to examine the documents that the system predicts are most relevant first.

A typical document collection will cover a wide variety of subjects. A user's request is likely to be relatively specific, so only a small fraction of the collection

will be relevant to the query. Furthermore, the non-relevant documents do not really define a class, since they cover all topics that aren't related to the query, which means that the typical pair of non-relevant documents are unlikely to have much in common. These properties of the IR problem suggest that a very large training sample will be necessary in order to: (1) find enough relevant documents and (2) get a reasonable description of the distribution of non-relevant documents.

There are a few factors which make the IR problem more tractable. First, there are only two categories, relevant and non-relevant, which simplifies the classification task considerably. Second, when the model is based on LSI generated feature variables, identification and interpretation of important features is not an issue. Classification accuracy is the only measure that is significant. Even interpretation of term-based classification rules is difficult, as they tend to require a large number of variables to work effectively.

The work in this thesis will focus on two primary methods, discriminant analysis and logistic regression. Discriminant analysis models each class with its mean vector and covariance matrix and assigns new observations to the class that they are most likely to belong to. Logistic regression estimates the probability of relevance under the assumption that the log ratio of the class-conditional densities is linear. These two methods will be described in detail in the next two sections. The last two sections will describe other methods which could be applicable to this problem.

6.2 Discriminant Analysis

It is common to approach the classification problem by assuming that the populations being analyzed can be modelled using the multivariate normal distribution. The mean vector and covariance matrix is then estimated for each group, and new observations are assigned to the group which they are most likely to belong to. This classification technique is known as quadratic discriminant analysis, since it partitions the response space along quadratic boundaries. Alternatively, the group covariance matrices can be assumed to be equal, a special case which is known as linear discriminant analysis, since it produces linear decision boundaries. However, linear discriminant analysis

can also be obtained using an alternative formulation, as described below.

6.2.1 The Discriminant Rules

The classical form of the two group linear discriminant analysis problem can be derived as follows. Suppose that one has a sample of data from two groups with mean vectors $\bar{x}_1$ and $\bar{x}_2$ and covariance matrices S_1 and S_2 respectively. The goal is to find the linear combination of the variables that maximizes the separation between the groups. A reasonable optimization criterion, as suggested by Fisher, is to maximize the separation between the means, scaling to reflect the pooled within groups covariance matrix. In other words, choose a so that:

$$a^* = \arg \max_a \frac{a^\dagger(\bar{x}_1 - \bar{x}_2)}{\sqrt{a^\dagger S a}}$$

is maximized, where $(n_1 + n_2 - 2)S = (n_1 - 1)S_1 + (n_2 - 1)S_2$.

Since S is positive definite, we can define the Cholesky decomposition of $S = R^\dagger R$. Let $b = Ra$, then the formula above becomes:

$$\arg \max_b \frac{b^\dagger R^{\dagger-1}(\bar{x}_1 - \bar{x}_2)}{\sqrt{b^\dagger b}}$$

which is maximized by choosing $b \propto R^{\dagger-1}(\bar{x}_1 - \bar{x}_2)$, which means then that $a^* = R^{-1}b = S^{-1}(\bar{x}_1 - \bar{x}_2)$. Therefore, the one dimensional space defined by $y = a^{*\dagger}x$ should cause the group means to be well separated. In most classification problems, new observations are classified as belonging to the group with the closest mean vector, perhaps scaling for an unequal loss function or the prior probabilities of class membership. However, for use in information retrieval, a simple ranking of the data is sufficient. This approach to linear discriminant analysis can be generalized to more than two groups [29].

If the two populations are assumed to be normal with a common covariance matrix, i.e.

$$f_k(x) = 2\pi^{-p/2} |\Sigma|^{-1/2} \exp\left[-\frac{1}{2}(x - \mu_k)^\dagger \Sigma^{-1}(x - \mu_k)\right]$$

then the log likelihood ratio of group membership is:

$$L = \log \frac{f_1(x)}{f_2(x)} = -\frac{1}{2}(\delta(x, \mu_1, \Sigma) - \delta(x, \mu_2, \Sigma))$$

$$\delta(x, \mu, \Sigma) = (x - \mu)^\dagger \Sigma^{-1} (x - \mu)$$

the quadratic components cancel and the rule reduces to:

$$L = x^\dagger \Sigma^{-1} (\mu_1 - \mu_2) + C_0$$

If μ_i and Σ are replaced by their unbiased estimates x_i and S , this is equivalent to the discriminant rule derived above [53]. The quadratic discriminant rule is simply:

$$L = -\frac{1}{2}(\delta(x, \mu_1, \Sigma_1) - \delta(x, \mu_2, \Sigma_2))$$

Either the linear or the quadratic rule can be used to induce a ranking of documents according to their likelihood of relevance, and the ranking can be compared to the results produced by traditional retrieval strategies.

6.2.2 Linear vs. Quadratic

Given a choice between linear and quadratic discriminant analysis, which method should be preferred? The first consideration should be the quantity of sample data. Let n_i be the number of observations in group i and p be the number of variables. If $n_i < p$, the covariance matrix of the group cannot be completely determined. If n_i is only slightly greater than p then the estimate of the covariance matrix will be extremely variable. In either of these cases, the linear discriminant rule is likely to be far more successful. If $n_i \gg p$, then the quadratic discriminant rule should work better, unless there is strong evidence that the group covariance matrices are equal. However, that still leaves a wide middle ground to be considered. In general the performance of the quadratic rule improves as heteroscedasticity increases and as the separation between groups decreases. The choice reduces to the traditional variance-bias trade-off. The linear rule reduces the variance in the parameter estimates, but at the expense of having biased estimates of the covariance matrices [53].

For problems in information retrieval, there will rarely be more relevant documents than variables, so quadratic discriminant analysis will generally be out of the question. Unfortunately, the assumption of homoscedasticity seems particularly poorly suited to information retrieval, where the relevant class is very precisely defined and the

non-relevant class is a grab bag of everything else in the collection. Fortunately, there are methods which do not force us to make a binary selection between linear and quadratic discriminant analysis.

6.2.3 Regularized Discriminant Analysis

It has been mentioned that when n_i is small relative to p , the covariance matrix S_i is poorly estimated. It is useful to examine the estimation problem in more detail. Using the spectral decomposition, a covariance matrix S_i can be represented as:

$$S_i = \sum_k \lambda_{ik} u_{ik} u_{ik}^\dagger, \quad S_i^{-1} = \sum_k \frac{u_{ik} u_{ik}^\dagger}{\lambda_{ik}}$$

where u_{ik} and λ_{ik} are the k 'th eigenvector and eigenvalue of S_i respectively. It is an unfortunate fact that the estimates λ_{ik} are biased. In particular, the larger eigenvalues are biased upwards and the smaller eigenvalues are biased downwards. A heuristic explanation for the largest and smallest eigenvalue is provided below. For more general results, see Cacoullos and Olkin [10].

The largest eigenvalue satisfies:

$$\lambda_1(X) = \max_a \frac{a^\dagger X a}{a^\dagger a}$$

A scalar function of matrix is defined to be convex if:

$$g(t * X + (1 - t) * Y) \leq t * g(X) + (1 - t) * g(Y)$$

Let $x_1 = \lambda_1(X)$, $y_1 = \lambda_1(Y)$, and choose a st $\lambda_1(tX + (1 - t)Y)$ is maximized, then:

$$\frac{a^\dagger X a}{a^\dagger a} \leq x_1 \quad \text{and} \quad \frac{a^\dagger Y a}{a^\dagger a} \leq y_1$$

which means that λ_1 is convex. Jensen's inequality, which states that $Eg(X) \geq g(EX)$ if g is convex, can be generalized to matrix functions [10]. Therefore, $E(\lambda_1(S)) \geq \lambda_1(\Sigma)$. A similar argument can be made for λ_p using the fact that the minimum is concave and Jensen's inequality is reversed for concave functions.

Since the discriminant rules are based on the inverse of the covariance matrix, this bias means that the importance of directions defined by the eigenvectors with

the smallest eigenvalues is greatly exaggerated. In order to mitigate this effect, it is desirable to obtain more reliable estimates of Σ_i . Regularization, introducing bias in the hope of reducing variance, is a reasonable approach to this problem [26]. Linear discriminant analysis represents one relatively extreme regularization method. One can imagine a much more reasonable set of regularization alternatives.

Friedman [26] introduces the concept of regularized discriminant analysis (RDA) by presenting a pair of shrinkage parameters which produce a very general family of estimators for Σ_i . Rather than simply choosing between the pooled and unpooled covariance matrices, one can consider using some weighted combination of them.

$$\hat{\Sigma}_i(\lambda) = \frac{(1 - \lambda)(n_i - 1)S_i + \lambda(n - g)S}{(1 - \lambda)(n_i - 1) + \lambda(n - g)}$$

$\lambda = 0$ yields QDA while $\lambda = 1$ describes LDA. However, λ can be chosen to be any value between 0 and 1.

While the regularized covariance matrix above gives a more general alternative to either LDA or QDA, it may be desirable to have even more regularization. If $n = \sum n_i$ is only slightly larger than p it may not even be possible to estimate the pooled covariance matrix accurately. In this case, it may be helpful to shrink the covariance estimate towards a very stable alternative, such as the identity matrix. If this is combined with the regularization described above, we obtain:

$$\hat{\Sigma}_i(\lambda, \gamma) = (1 - \gamma)\hat{\Sigma}_i(\lambda) + \frac{\gamma}{p}\text{tr}(\hat{\Sigma}_i(\lambda))I_p$$

The regularization parameter γ decreases the estimates of the largest eigenvalues and increases the estimates of the smallest eigenvalues, thus countering the bias inherent in S_i . When combined, these parameters produce a relatively large family of estimators that should be suited to a wide variety of situations. Friedman [26] presents a series of simulations that show that regularized discriminant analysis performs at least as well as LDA and QDA and usually considerably better for a number of different distributions for the group covariance matrices.

In order to use RDA, the two regularization parameters must be estimated. Friedman estimates the optimal parameter settings by jointly minimizing the cross-validated error rate over a grid of values. He derives updating formulae which allow

the regularized estimates to be computed relatively efficiently. It turns out that removing an observation from the covariance matrix is equivalent to down-dating it by a rank-one matrix plus a multiple of the identity. Both of these operations can be accomplished without reinverting the covariance matrix. Additional savings come from the fact that for fixed λ the eigenvectors are independent of γ and thus do not need to be recomputed to update by different multiples of the identity matrix.

Regularization seems like an excellent idea for the IR problem. There are too few relevant documents to properly estimate Σ_r , but it is unlikely that $\Sigma_r = \Sigma_n = \Sigma$. Regularization towards quadratic discriminant analysis would allow us to use some of the information in S_r without being too heavily penalized for its large variability. Even after dimension reduction using LSI, there are still 100-200 variables in the model, so regularization towards the identity may also be helpful. Unfortunately, the size of the problem is such that optimization over a two-dimensional grid of parameter values using cross-validation will not be feasible, particularly on a query by query basis. Instead it will probably be necessary to remove a validation set from the original training set and use it to select the optimal regularization parameters.

6.3 Logistic Regression

Regression is a general term in statistics used to describe a variety of techniques for modelling a response variable using one or more predictive variables. Linear regression, which models the response by a linear combination of the predictors, is the simplest and most widely used of these methods. In information retrieval, we want to use regression to estimate the probability of relevance of each document, taking the LSI factor scores of the document as its predictor variables. Unfortunately, linear regression assumes that the response variable is continuous and supported on the whole real line, while our data is binary in nature (e.g. 1 = relevant, 0 = not relevant).

While linear regression could be applied to our data, it would produce a poor model due to its least squares optimization criterion. Linear regression minimizes the sum of the squared differences between the observed and predicted values. If the

response variable is binary, least squares penalizes very high scoring and very low scoring documents in a misguided effort to squeeze their predicted values as close to zero or one as possible. In order to avoid this behavior, the predicted values must be allowed to vary over the whole real line and then transformed in such a way that the probability estimates will fall on the unit interval.

The logistic link function: $g(\pi) = \log(\pi/(1 - \pi))$ is one function that has this property. Fitting a model of the form:

$$g(\pi_i) = \eta_i = \sum_j x_{ij} \beta_j$$

is known as logistic regression. Using logistic regression, we can estimate the β_j 's using the training data and apply the resulting model to estimate the probability of relevance of each document in the test data. The documents can then be ranked and the results compared to other methods using the traditional evaluation measures.

The parameters of the logistic model can be estimated using maximum likelihood [52]. Each response y_i is assumed to come from an independent binomial random variable with n_i trials and a probability of relevance π_i . Note that $n_i = 1$ for all i , since no two documents have the same distribution of term or LSI factors. The log of the likelihood function (ignoring constants) has the form:

$$L(\pi; y) = \sum_i [y_i \log(\frac{\pi_i}{1 - \pi_i}) + \log(1 - \pi_i)]$$

We can maximize the likelihood by setting the partial derivatives $\delta L / \delta \beta = 0$, which is accomplished using several iterations of the chain rule:

$$\frac{\delta L}{\delta \beta_k} = \sum_i \frac{\delta L}{\delta \pi_i} \frac{\delta \pi_i}{\delta \eta_i} \frac{\delta \eta_i}{\delta \beta_k}$$

yielding the result that:

$$\frac{\delta L}{\delta \beta_k} = \sum_i (y_i - \pi_i) x_{ik}$$

Since the likelihood equations are nonlinear in β , they must be solved by an iterative method. Given $\hat{\beta}_0$, we can compute $\hat{\eta}_0$ and $\hat{\pi}_0$ and estimate $\hat{\beta}_1$ using Newton-Raphson:

$$-\frac{\delta^2 L}{\delta \beta^2}(\hat{\beta}_1 - \hat{\beta}_0) = \frac{\delta L}{\delta \beta}$$

We find that:

$$-\frac{\delta^2 L}{\delta \beta_k \delta \beta_l} = \sum_i \pi_i (1 - \pi_i) x_{ik} x_{il}$$

If we define W to be a diagonal matrix with entries $w_{ii} = \pi_i(1 - \pi_i)$, then we can rewrite the Newton-Raphson equation as:

$$X^\dagger W X \hat{\beta} = X^\dagger W z \quad \text{where} \quad z_i = \eta_i + \frac{y_i - \pi_i}{\pi_i(1 - \pi_i)}$$

Therefore, the equation reduces to a weighted least squares problem, which can be solved iteratively using standard regression techniques.

Logistic regression suffers from some of the same problems as linear discriminant analysis. It requires considerably more data points than variables to obtain accurate estimates of the parameters and the fitted values. The small sample of relevant documents will make it extremely sensitive to relevant documents that are outliers in the parameter space. Furthermore, it requires considerably more time to compute than LDA since the weighted covariance matrix must be inverted during each iteration of least squares as opposed to just once for LDA. This becomes a concern with the large data matrices used in information retrieval problems. Logistic regression does bring along a large suite of diagnostic tools for estimating the significance of the parameters, but since we are interested only in the predictive power of the model, these tools do not have a lot of value in information retrieval. However, it is worthwhile to have a reasonable alternative to discriminant analysis.

6.4 Optimal Separating Hyperplanes

Both LDA and the logistic model are linear classification methods because their decision rules can be expressed as a linear combination of the predictor variables. This is equivalent to saying that the classification boundary between two groups is given by a hyperplane in the predictor space. An alternative classification strategy is to find the optimal separating hyperplane directly by minimizing some measure of the error associated with the misclassified observations. One such approach is to solve

the following optimization problem [5]:

$$\frac{1}{m} \sum_{i=1}^m (-a_i^\dagger \omega + \gamma + 1)_+ + \frac{1}{n} \sum_{j=1}^n (b_j^\dagger \omega + \gamma + 1)_+$$

where a_i and b_j are the feature vectors for the observations from the two groups, ω is a weight vector, and γ is a constant.

The solution to this problem will generate the plane $\omega^\dagger x = \gamma$ which minimizes the average L_1 distance of any points which lie on the wrong side of the plane from the majority of the group. The additional factor of one in the summations is arbitrary and could be replaced by any positive constant. It is included to screen out the degenerate optimal solution $\omega = 0$, $\gamma = 0$. Note that penalties are scaled by the size of the group so violating points from the smaller group are penalized more heavily.

Since the L_1 -metric is used to accumulate the errors, the optimization problem can be re-expressed as a linear program and solved using classical techniques such as the simplex method. The linear program is defined as follows:

$$\min_{\omega, \gamma, y, z} \left\{ \frac{ey}{m} + \frac{ez}{n} \mid A\omega - e\gamma + y \geq e, -B\omega + e\gamma + z \geq e, y \geq 0, z \geq 0 \right\}$$

where A and B are data matrices for the two groups and $e = [1, \dots, 1]$. Our experiments using this method relied on the Minos numerical optimization package obtained from the Systems Optimization Laboratory at Stanford University [56].

In practice, this optimization strategy was not found to be particularly effective for the information retrieval problem. The location of the separating plane tended to depend too much on the location of one or more violating relevant documents, whose penalty terms were heavily magnified by the averaging process. By concentrating only on the boundary conditions, the technique does not attempt to capture the structure of the correctly classified relevant documents, which may contribute to its poor performance for the information retrieval problem.

6.5 Classification Trees

The classification problem can be approached in a completely different manner by creating a classification rule that is based on a binary tree. This method is often

known as recursive partitioning or classification trees [7]. At each node in the tree, the system decides between the two sub-branches by examining a binary decision rule that is generally based upon a single variable (e.g. $x_i > c$). The system derives the binary decision rule at each node by maximizing a purity measure of the two subsets over all possible binary splits that could be applied to the predictor variables. Splitting continues until the terminal nodes are pure (all elements are from the same group) or until the node falls below the default size threshold.

The initial splitting procedure produces a nested sequence of trees, where the first tree consists only of the root node and each subsequent tree adds splits along every eligible terminal node of the preceding tree in the sequence. Since the final tree will almost certainly overfit the training data, the goal is to determine how many branches (splits) must be pruned from the complete tree to produce the classification tree that will be most successful at categorizing new data. The optimal tree is chosen by cross-validation using a cost complexity function that balances the estimated error rate and the size of the tree.

Classification trees have been previously applied in information retrieval with mixed success. Lewis and Ringuette [49] apply it to the text categorization problem on the Reuters collection and achieve reasonably good results. Tong et. al. [78] apply it to the routing problem on the TREC corpus where it is not nearly as successful. Both implementations use words as features. The different levels of performance can be traced to differences in the problem and in the test collections. Wiener [82] reports that a large fraction of Reuters topics can be well classified using a very small number of terms. For example, for the topic labelled “Wheat”, it should not be surprising that presence of the word “wheat” alone serves as a very strong indicator of class membership. The TREC topics defy such simple categorization and require much more complex combinations of terms.

Tong’s results are quite similar to our own. Classification trees did not prove to be very successful in any of the routing applications on which they were tried. The primary reason appears to be the fact that they split along a single variable. It quickly became clear that the relevant region for typical queries in information retrieval depends on combinations of large numbers of variables. Classification trees

are not very successful for these problems, because they must split on one variable at a time. In addition, they do not provide a complete ranking of the documents in the collection, since elements in a terminal node have the same score. This complicates the problem of evaluation.

The advantages of the classification tree approach are its ability to pick up unusual interactions among the variables and the easy interpretation of the classification rule. Neither of these advantages prove to be particularly useful when LSI factors are used as feature variables. The LSI factors are constructed to be orthogonal, so the chances of an important interaction in the model are significantly reduced. Once the features are transformed into factor space, any hope of simple interpretation of the classification rule is lost. While it is possible to apply classification trees using splits along linear combinations of the variables, these methods are not computationally feasible for the size of training sets used for the information retrieval problem.

Other methods that could be considered include neural networks and a host of other flexible non-parametric methods for data classification. These methods are not examined in this thesis for several reasons. First, they tend to be computationally intensive, so it would be difficult to train them on large IR data sets in a reasonable amount of time. Second, the key advantage of these methods is their ability to capture interactions and non-linear structure. The large number of feature variables makes identifying important interactions difficult and it is unlikely that the data in information retrieval is of high enough quality (due to the small number of relevant documents) for a classification rule to be able to capture nonlinear structure. Schuetze [73] has achieved very good results using neural networks to solve the routing problem, however he reports that performance gains from using a nonlinear component in the network are very small. Our experimental work, which will be presented in the two following chapters, will be based primarily on discriminant analysis and logistic regression.

Chapter 7

Retrieval Experiments: Cranfield

In previous chapters, we have described the standard approaches to the information retrieval problem and some of the attempts to improve upon the basic model. As discussed in chapter 5, computer simulation experiments are a good way to objectively compare the performance of different retrieval strategies. These experiments are based on a predetermined set of queries which have known relevant documents. This chapter will present results from a series of retrieval experiments conducted on the Cranfield collection.

The IR problem can be divided into three categories: adhoc search, relevance feedback, and routing. In the adhoc search problem, the system analyzes the query and converts it into a mathematical representation (e.g. a vector of weighted term frequencies) which is then used to rank the documents in the collection. The information available to the system is limited to the text of the query, so improving performance using statistical methods is relatively difficult. Better results can be obtained primarily by finding an improved representation of the query.

In relevance feedback, it is assumed that the user will examine some of the highest ranking documents, determine if they are relevant, and submit this information to the system for use in a revised query. Once again, information is limited but it is possible that a sequential classification algorithm might provide improved performance when compared to the standard model. In routing, the system can draw upon a significant sample of previously evaluated documents when constructing the query. Routing

is identical to the statistical classification problem and holds the greatest potential for obtaining improved results using more complex classification algorithms. The majority of the experimental work presented here will be directed towards obtaining better solutions to the routing problem.

This chapter will present the results of a series of computer experiments. It will begin by describing the Cranfield collection, presenting summary statistics to provide a picture of the word frequency distributions within queries and documents. It will then examine the impact of term weighting and document normalization on performance, with the goal being to provide a measure of the sensitivity of evaluation measures to these issues. Subsequent experiments will produce baseline results for relevance feedback and routing based on the Vector Space Model.

Since the information content of individual terms is relatively limited and a huge number of terms are available to serve as predictors, it is vital to perform some degree of dimension reduction before turning to a statistical classification algorithm. As described in chapter 4, Latent Semantic Indexing (similar to Principal Components Analysis but based on the singular value decomposition) shows the most promise among the potential dimension reduction strategies available for IR problems. Therefore, in the next experiment we compute the LSI solution for the Cranfield collection and examine its performance with respect to the baseline values. Finally, we make an initial attempt to improve performance for routing by applying a classification algorithm.

7.1 The Cranfield Collection

The Cranfield corpus is a collection of abstracts of journal articles on the subject of aeronautical engineering. The version used in these experiments contains 1397 documents and 225 queries. The documents have 6174 unique terms, after removing a stop list of frequent terms and applying a stemming algorithm developed at Xerox PARC. All subsequent details presented here are based on this processed version of the original collection.

The Cranfield collection was selected as the basis for our initial retrieval experiments for a number of reasons. The Aslib-Cranfield project, conducted in the 1960's, was one of the first attempts to test the performance of IR systems using computer experiments, and it was based on this collection [12]. Since then, Cranfield has been used in countless retrieval experiments and represents one of the standard collections for evaluation in the IR community. It also is a small enough collection that system processing time will not become an issue during the course of the experiments. Furthermore, it has a large enough sample of queries that any observed differences in performance are likely to be statistically significant.

Before moving on to the actual experiments, it is helpful to present some summary statistics that characterize the distribution of terms over queries and documents. The table below gives the distribution of term frequencies and counts over the collection:

terms	freq.count	doc.count
1	2200	2458
2-4	1573	1600
5-10	847	785
11-20	474	457
21-50	505	450
51-100	248	212
101-200	166	131
201-500	117	74
501-1000	37	7
1001-2000	6	0
2001-5000	1	0

The column labelled freq.count displays a histogram of the number of terms that occur within the given frequency range over the collection. The doc.count column displays a histogram of the number of terms that occur within the specified number of documents over the collection. The heavily skewed distribution is typical for terms in language.

Figure 7.1 shows histograms of the distribution of document lengths, query lengths, and the number of relevant documents per query. The average document length is

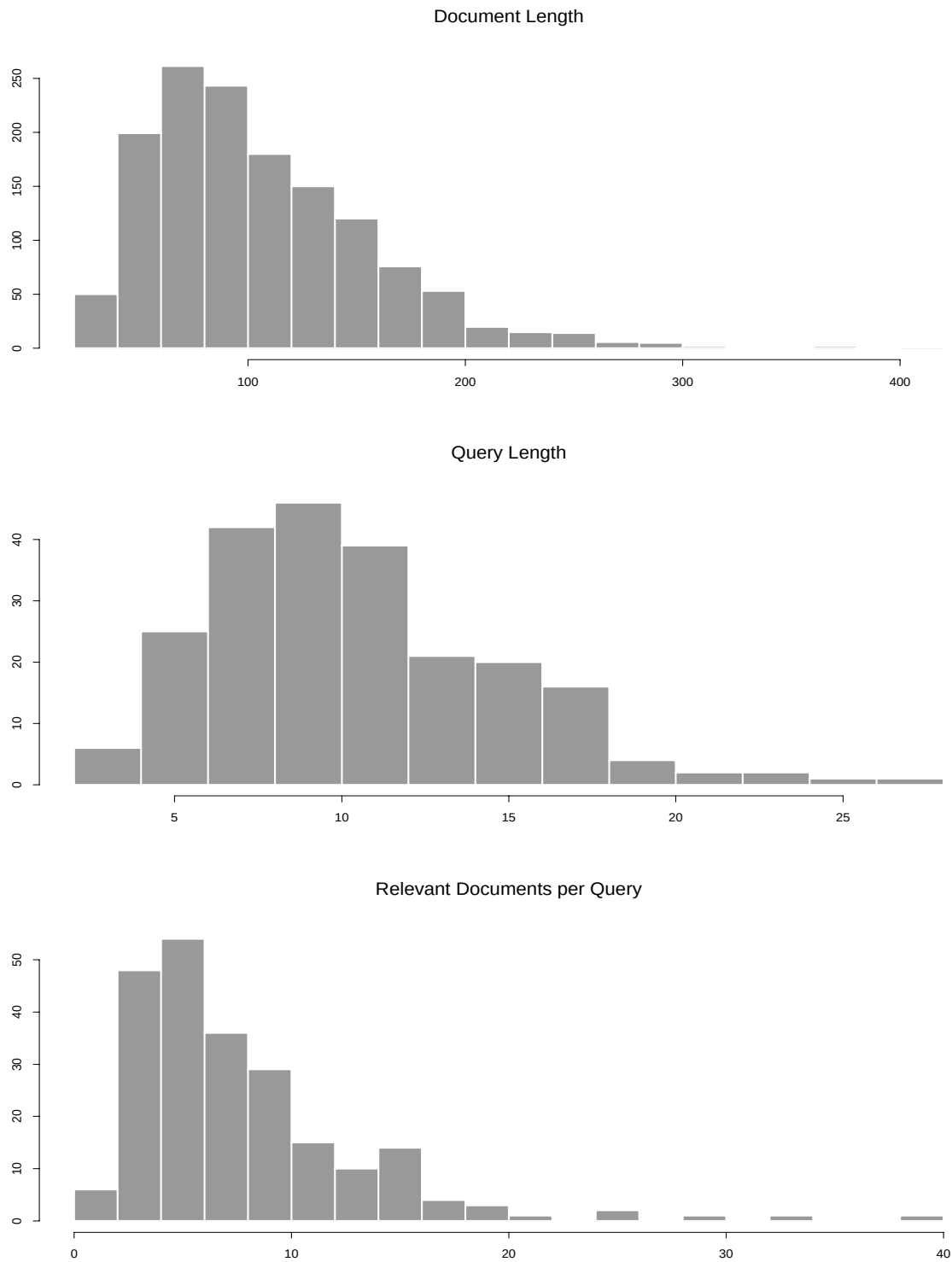


Figure 7.1: Cranfield: Summary Statistics

106 terms. Since the documents are abstracts, they tend to be only one or two paragraphs long. The average query length is 11 terms. Most of the queries are only one sentence long. A sample query is shown below to give the reader an idea of the style and content of queries:

How do interference-free longitudinal stability measurements (made using free-flight models) compare with similar measurements made in a low-blockage wind tunnel?

Both of these histograms are based on the document and query representation after stemming and stop list removal. There are an average of only eight relevant documents per query, ranging over queries from as few as two to as many as 40 relevant documents. The small number of relevant documents will cause problems, particular for the routing task, and represents one of the hazards of dealing with a small collection.

7.2 Baseline Experiments using the VSM

Before any improved retrieval models can be analyzed, it is important to establish a baseline level of performance against which other strategies can be compared. Since the Vector Space Model is probably the most common benchmark in information retrieval and it typically performs well under a wide variety of conditions, it will be used to compute the baseline level of performance for query-based search, relevance feedback, and routing. This will also verify that the search routines written by the author are accurate and can attain a level of performance comparable to the results obtained in other research experiments on this collection.

This section begins with a brief discussion of the author's implementation of search routines using the Vector Space Model. It then describes a series of retrieval experiments which produce baseline results for the three retrieval problems discussed above.

7.2.1 Implementation of Vector Space Search

As described in chapter 3, the Vector Space Model represents queries and documents by vectors of weighted term frequencies and estimates the similarity between them by the inner product of their corresponding vectors. In practice, however, the document scores are not computed in this fashion, since it is inefficient to read in the full term vector representation of every document in the collection to conduct a search. Rather, the document scores are computed term by term using the inverted index.

The search is initiated by creating two vectors of length equal to the number of documents in the collection with each entry initialized to zero. Then, the system looks up the first query term in the inverted index to determine which documents contain that term. The product of the query weight and the document weight for that term is then added to the document entry of the score vector. The second vector is used for normalization and accumulates the document weights alone. This process is repeated for each term in the query. At the end, the document score vector is divided by the document normalization vector to obtain the final document scores. A diagram of this process is given below:

- 1) Initialize a document vector D and norm vector N .
- 2) Compute the query weight vector $Q = \{q_i\}$.
- 3) For each term i in the query, do:
 - (a) For each posting j in the index entry of term i , do:
 - (b) Compute weight d_{ij} and add normalization factor $n(d_{ij})$ to entry n_j of N .
 - (c) Compute $q_i * d_{ij}$ and add it to entry d_j of D .
- 4) Set $d_j = d_j/n_j$ to produce the final normalized score vector.

The score vector is then used to compute the ranks of the relevant documents. Since all relevant documents are considered equally valuable, there is no need to differentiate between them. A full ranking of the document scores is not required. Rather, we can proceed as follows:

- 1) Rank the relevant documents in order from lowest similarity d_1 to highest similarity d_k .
- 2) Initialize a rank vector $R = \{r_i\}$ of length equal to the number of relevant documents. The highest scoring relevant document receives an initial rank of 1, the 2nd highest an initial rank of 2, etc. . . .
- 3) For each non-relevant document score n_j , do:
 - 4) Let $i = 1$.
 - 5) If $n_j > d_i$, $r_i \leftarrow r_i + 1$, $i \leftarrow i + 1$, repeat (5), else $j \leftarrow j + 1$, go to (4).

The final rank vector R contains the ranks of the relevant documents and can be passed on to an evaluation algorithm. For the experiments conducted here, evaluation is based primarily on precision averaged at 10% through 100% recall, for two reasons. First, this makes it easier to compare our results with experiments by other researchers on this collection, which tend to use a similar measure. Second, we will be conducting a large number of retrieval experiments and so will not have the time to carefully analyze every result using more than one retrieval measure. The average of the precision-recall curve is probably the best single measure to reflect performance over a wide variety of recall levels.

7.2.2 Term Weighting and Basic Search

Using the algorithms defined above, we can now implement search strategies based on the Vector Space Model. The first experiment examines the most important factor that influences search performance, the choice of term-weighting strategy. As described in chapter 3, query and document weights are usually based on the following weight functions, using i as the term index:

$$\text{Query}(Q): q_i = \text{TF}(qtf_i) * \text{TW}_i \quad , \quad \text{Document}(D_j): d_{ij} = \text{TF}(dtf_{ij}) / \text{N}(D_j)$$

where qtf is the query frequency, dtf is the document frequency, TF is the term frequency weight, TW is the term importance weight, and N is the normalization function.

There are three goals in this experiment: (1) find reasonable weighting functions TF, TW, and N for use in future experiments, (2) establish a baseline performance level for query-based search, and (3) get some perspective on the degree of variability associated with choosing different term weights. Attention will be restricted to a very limited set of weighting options that have been found to work well in previous term-weighting experiments. The initial run examines the following options for term frequency and term importance weights. All documents are normalized to have unit length based on the L_2 -norm.

TF1:	$\log_2 \text{tf}$	TW1:	$\log_2 N/n_i$
TF2:	$\sqrt{\text{tf}}$	TW2:	$(\text{TW1})^2$
TF3:	tf	TW3:	$\log_2 N - \sum p * \log p$
TF4:	if (tf > 0) 1 else 0	TW4:	$(\text{TW3})^2$

tf = term frequency (in query or document) , $p = \text{tf} / \sum \text{tf}$

$n_i = \# \text{ docs containing } i$, $N = \# \text{ docs in collection}$

It is generally accepted that the importance of a term increases at a rate no greater than its frequency (TF3). The opposite extreme is to ignore frequency entirely and use binary weights (TF4). Two intermediate transformations of the data are also tested. The entropy or S/N ratio (TW3) and the inverse document frequency or IDF (TW1) are described in chapter 3. The IDF score is derived from the entropy score using binary term frequencies. The square of these weight functions corresponds to using importance weights for both the query and the document. Since TW is constant over documents, it is more computationally efficient to incorporate it into the query weight.

The experiment will examine all combinations of the factors above. The queries in Cranfield are short enough to permit a large number of experimental runs. The results can be analyzed using a three-way ANOVA [51], modelling TF and TW as fixed effects and the Query as a random effect. The interaction between TF and TW will also be tested.

	Df	Sum of Sq	Mean Sq	F Value	Pr(F)
Term.Freq	3	7.363	2.4542	65.898	0.000
Term.Wt	3	0.399	0.1330	22.704	0.000
Query	224	241.579	1.0785		
Term.Freq:Term.Wt	9	0.042	0.0046	1.761	0.071
Term.Freq:Query	672	25.027	0.0372		
Term.Wt:Query	672	3.936	0.0059		
Residuals	2016	5.285	0.0026		

Clearly, the choice of TF and TW both influence performance. The magnitude of the interaction term is very small in comparison, so it is fair to assume that there is no interaction between TF and TW. A table of the average precision-recall scores by weight category is given below:

	TW1	TW2	TW3	TW4
TF1	0.355	0.364	0.349	0.356
TF2	0.367	<u>0.374</u>	0.364	0.369
TF3	0.345	0.356	0.333	0.345
TF4	0.298	0.310	0.288	0.299

Using the square root of term frequency gives the best performance, but log and raw frequency are fairly close. The only clear loser is binary weights. It makes sense to use the square root transformation from several other perspectives. First, the L_2 -norm of the square root transformed data is simply the square root of the document length, which has been pre-computed for all documents. Second, the term frequencies are count data, and while they don't exactly follow the poisson distribution, it is probably the best first-order approximation. The square root is well known as the variance stabilizing transformation for the poisson distribution [6], and equal variance is a desirable property for term weights.

The IDF performs slightly better than the entropy measure, and using the square of the factors is marginally preferred. The practical significance of these results is limited. It might seem that since the entropy weights use more information than the IDF that they should produce better results. However, the documents are short

enough relative to the frequency of the terms that the probabilities used in the entropy calculation probably cannot be estimated accurately. Since the IDF is much easier to compute, it is definitely preferred.

A similar experiment tested the impact of several different options for the normalization function, and the results (not detailed here) led to two conclusions. First, the L_2 -norm performs better than the L_1 -norm. Second, it is better to normalize documents before multiplying by the term importance factor, rather than after. This makes some intuitive sense. One would not want the term weights of the rest of the document to drop sharply when a few unique terms are added to the document. The L_2 -norm is also the easiest to compute under the square root frequency transformation, as described above.

Based on these results, the following term weighting scheme has been chosen for subsequent experiments:

$$\text{Query}(Q): q_i = \sqrt{(qtf)_i * \log N/n_i^2} \quad , \quad \text{Document}(D_j): d_{ij} = \sqrt{\frac{(dtf)_{ij}}{\sum_j (dtf)_{ij}}}$$

$$\text{Sim}(Q, D_j) = \sum_i q_i * d_{ij}$$

To put these results in perspective, we can compare them to other experiments on the same collection. The authors described below use average precision at 25%, 50%, and 75% recall. The score of our weighting methods described above is 0.386 under this measure. Experiments by Harman [33] and Salton [69] obtain the values 0.146 and 0.116 respectively for the same collection. It is hard to explain why these results are so much worse. Perhaps the other researchers were using a different version of the collection, for it is nearly impossible for a difference of this magnitude to be accounted for by an alternative choice term weights, stop list, or stemming algorithm. However, Aalbersberg [2] obtains a score of 0.390, which is right in line with our results. This indicates that our system is working as designed and obtaining values comparable with the best results obtained by other authors.

Note that no claim is being made that these weight functions are optimal in any sense. The experiment above examines a very limited family of weighting strategies over a small document collection on a very specialized topic. However, other

researchers have found that this class of weighting functions performs robustly over a number of different experimental test collections.

7.2.3 Baseline: Relevance Feedback

Relevance feedback improves performance by incorporating the text from relevant documents identified by the user into a revised query. Our experiments will use a variant of the Rocchio method described in chapter 3:

$$Q_{rfb} = k * Q_0 + \sum_{i \in rel} \frac{D_i}{|D_i|} - d * \sum_{j \in nrel} \frac{D_j}{|D_j|}$$

This strategy adds the sum of the relevant documents and subtracts a weighted sum of the non-relevant documents from the query. The following parameters can be adjusted in this model:

- 1) The number of documents examined between each iteration of relevance feedback.
- 2) The importance of the query (k).
- 3) The weight of non-relevant documents (d)
- 4) The number of terms to use in feedback experiments (n).

A revised query is computed for each iteration of relevance feedback and the query is then compared to all documents that haven't yet been evaluated. Aalbersberg [2] found that the best performance could be obtained by performing an iteration of feedback for each new relevant document that is discovered. He conducted an exhaustive set of experiments on the Cranfield collection, so there is no reason for us to duplicate his effort. Therefore, we will simply adopt this strategy.

The query importance weight will determine the stability of the query during the initial iterations of the feedback process. It can be thought of as a prior estimate of our confidence in the query. If it is large, then the query will only change slightly during each iteration. If it is small, then the revised query will be dominated by the distribution of the evaluated documents. It is also interesting to measure the influence

of the non-relevant documents and determine whether they have a significant impact on performance.

There are several reason why it might be helpful to be selective in adding terms from the newly evaluated documents. Harman [33] found that the best performance during feedback could be obtained by carefully selecting a small number of terms to add to the query. Furthermore, relevance feedback is an expensive process, so if good results can be achieved with a lot fewer terms, this is a considerable advantage. We rank terms according to their query weights as given by the current feedback query and truncate the query after n terms. Note that the list is dynamic and terms can enter or leave the list as their significance with respect to the query changes.

We will examine the following values of the parameters:

$$k = 2, 6, 10 \quad d = 0, 0.2, 0.4 \quad n = 20, 60, 100$$

Since relevance feedback is considerably more expensive than a simple query search, we will not look at all combinations of the parameter settings. Rather, we will use an orthogonal array to select nine combinations of the parameters such that the design is balanced in all two-dimensional margins [58]. The layout and its results are presented below:

k	d	n	v.k	v.d	v.n	p.r
10	0	60	1	-1	0	.489
10	0.2	20	1	0	-1	.462
10	0.4	100	1	1	1	.481
2	0	100	-1	-1	1	.491
2	0.2	60	-1	0	0	.482
2	0.4	20	-1	1	-1	.445
6	0	20	0	-1	-1	.465
6	0.2	100	0	0	1	.493
6	0.4	60	0	1	0	.480

where v.k, v.d, and v.n are the standardized values of k , d , and n that will be used to analyze the experimental results.

Since all of these parameters are continuous, we can fit a quadratic response surface with no interactions to the data to estimate the parameter setting that will obtain optimal performance.

```
Call: lm(formula = p.r ~ v.k + v.k^2 + v.d + v.d^2 + v.n + v.n^2)
```

Coefficients:

	Value	Std. Error	t value	Pr(> t)
(Intercept)	0.4888	0.0013	362.8152	0.0000
v.k	0.0025	0.0006	3.9822	0.0577
I(v.k^2)	-0.0041	0.0011	-3.8422	0.0616
v.d	-0.0064	0.0006	-10.2896	0.0093
I(v.d^2)	-0.0035	0.0011	-3.2867	0.0814
v.n	0.0153	0.0006	24.6148	0.0016
I(v.n^2)	-0.0108	0.0011	-9.9526	0.0099

Residual standard error: 0.001528 on 2 degrees of freedom

Multiple R-Squared: 0.9977

Since the quadratic terms are all negative the surface has a maximum value. It occurs at $k = 7.2$, $d = 0.02$, $n = 88$. Therefore, the surface is maximized (approximately) by giving the query a weight of seven relevant documents, assigning a non-relevant document 2% of the weight of a relevant document, and keeping only the 88 most important terms. Since the value of d is so small, we will simply set it to zero, since ignoring the non-relevant documents saves a great deal of computing time. When relevance feedback is run again, the final performance value obtained is 0.496. This matches very closely with the value of .499 obtained by Aalbersberg.

7.3 Baseline: Routing

The routing query benefits from drawing upon a large sample of previously evaluated documents. In an ideal routing experiment, the corpus would be divided into a training set of known relevant and non-relevant documents and a test set of unknown

documents to use for evaluation. However, with an average of only eight relevant documents per query, the Cranfield collection does not have enough relevant documents for this strategy to be feasible. We can still conduct a reasonable analysis by using the technique of cross-validation. The strategy is to remove one document at a time from the collection, and then use the remaining documents to try to predict the relevance of the missing document.

Given the results of our experiments using relevance feedback, it seems reasonable to represent the routing query by the sum of the relevant documents. It is particularly convenient to ignore the non-relevant documents in this context, since then cross-validation need only be performed over the relevant documents. Since relevant documents are a tiny fraction of the collection, this reduces the computational cost of the experiment substantially.

Cross-validation differs from using the mean of the relevant documents as the query only in the following manner. Each relevant document is compared to the mean of the remaining relevant documents after excluding the document being examined. Therefore, a relevant document cannot favorably influence its own performance. The different queries are normalized to have equal length, and documents are ranked according to their inner product similarity to the routing query. Performance is evaluated by the average precision at ten levels of recall, as in the previous experiments. We did not examine any variations of the basic model, so we conducted only a single routing experiment, obtaining a baseline value of 0.509. It is not a surprise that routing performs better than relevance feedback, since nearly all the useful relevance information is available initially, but one might expect an even greater improvement. This issue will be addressed in more detail in the following section.

7.4 LSI Experiments

Latent Semantic Indexing is a dimension reduction strategy that replaces terms with a much smaller number of orthogonal indexing variables. It is crucial to perform some amount of dimension reduction before considering various statistical classification methods. We hope to find that using an LSI representation of documents does not

lead to a substantial reduction in performance. The major issue to examine is the optimal number of singular vectors to retain in the solution. The experiments in this section will examine this question for the three search strategies analyzed above.

7.4.1 Description of Method

In order to match the LSI representation of the collection with the term representation, we apply the same term weighting method used above to the document by term matrix before computing the singular value decomposition. In addition, all terms of frequency one are eliminated from the matrix, since this reduces the number of terms by a factor of one third while losing a minimal amount of information. Some of these terms will later be factored back in to the model. The singular value decomposition is computed using a sparse matrix SVD program known as SVDPACKC [21]. Using a sparse matrix algorithm results in a considerable savings in time and space over traditional SVD algorithms.

The document by term matrix X is then decomposed into:

$$X = DST^{\dagger}$$

where D and T represent documents and terms respectively in the new coordinate space and S is a non-negative diagonal matrix of singular values. After the matrix is factored, only a subset of the singular vectors with the largest singular values are retained. The sparse matrix algorithm can efficiently estimate a number of factors that is no greater than a quarter of the dimension of the data matrix. Based on this limit and space considerations, it was decided to retain the largest 200 factors. See figure 7.2 for a plot of the singular value scores. The first singular value, with a score of 58.8, is deleted in order to give the plot a better perspective. Since all elements of the original data matrix are positive, the first singular vector can be thought of as a weighted mean of the observations, and all the elements of this vector are also positive.

Researchers experimenting with Principal Component Analysis have often advocated using this plot to determine how many factors to retain, looking for breaks or gaps in the curve as indicators that the following dimensions are much less important.

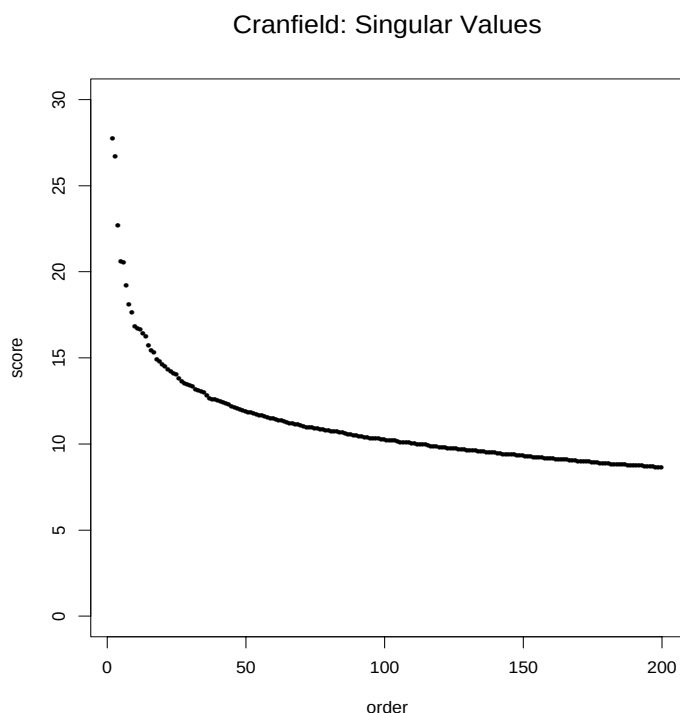


Figure 7.2: Cranfield: Singular Value Scores

Clearly no such technique will be useful in this context, as the curve decays smoothly after the first few factors. Furthermore, this decay is relatively slow after the first few factors, indicating that later factors still retain a fair amount of information. Therefore, we will have to rely on information retrieval experiments to determine an optimal number of factors.

LSI produces a 1397 by 200 document matrix D and a 3974 by 200 term matrix T . Each document and term is described by a 200 dimensional vector, and documents and terms are represented in the same subspace. Each document is a weighted combination of the vectors of the terms that it contains and vice-versa. Therefore, the query can be represented in LSI space by the weighted sum of its term vectors.

Unique terms that occur in one or more queries are represented in a similar fashion. The similarity between a query and the documents can be computed as follows:

$$\text{sim}(Q, D_j) = \sum_i q_i * s_i * d_{ij}$$

where q_i and d_{ij} are the weights for the i 'th LSI factor for the query and document j respectively, and s_i is the i 'th diagonal element of S . Adhoc search, feedback, and routing can be conducted using this new document representation.

7.4.2 Experimental Results

For adhoc search, the LSI representation of the query is constructed by taking the weighted sum of the term vectors, where the weights are chosen according to the term weighting formula developed in the previous section. The basic term model is compared to the LSI model using varying numbers of LSI factors. In all cases, the factors chosen are those with the highest singular values. In the routing problem, the results are obtained by cross-validation using a query that consists of the mean of the relevant document vectors in LSI space.

	terms	40	80	120	160	200
search	0.374	0.284	0.355	0.385	0.390	0.396
rfb	0.496	0.307	0.398	0.435	0.450	0.461
routing	0.509	0.406	0.507	0.544	0.556	0.567

The terms column gives the results of the baseline experiments using the term representation of documents as detailed in the previous section. The numbers indicate the number of LSI factors used in the model.

The results are analyzed by comparing each column to the baseline using the paired t-test and the Wilcoxon test, as described in chapter 5. For both search and routing all models with 120 or more factors are significantly better than the baseline. All models using relevance feedback are significantly worse than the baseline. Statistical tests used for comparing more than two methods, such as the ANOVA, are not employed because of the correlation inherent in the methods. For example, the

scores using the top 200 LSI factors are strongly related to the scores using the top 160 LSI factors.

From these results, we find that Latent Semantic Indexing is a remarkably successful dimension reduction technique. For search and routing, it can actually be used to improve retrieval performance, indicating that there is a lot of noise which can be eliminated from the data. Performance increases rapidly with the number of factors, reaching the level of the term based model after about 100 factors. It continues to improve up to 200 factors, but at a much slower rate. Note that eventually, the performance curve must peak, because it then declines to match the baseline when all factors are included in the model.

Unfortunately, LSI does not perform as well for relevance feedback, which should not come as a complete surprise. Both feedback and LSI perform some sort of variable selection. LSI selects the factors that are most important for capturing the global term correlation structure. Relevance feedback, on the other hand, retains all the information and selects the terms that are most valuable for predicting relevance for the query being analyzed. This ability to optimize variable selection at a local level is a considerable advantage. The LSI representation is probably losing information that is important in the region of the query, but far less significant on the global level. Note that the gap between search and feedback increases as more LSI factors are included in the model.

It is a bit surprising that LSI is so successful for routing given its problems with relevance feedback. There are a couple of factors that might explain this result. First, there is a lot more information available for routing, which may make up for important detail that is lost in a few of the relevant documents. Second, the baseline routing result uses no variable selection, retaining all the terms in the relevant documents. In hindsight, it might have been better to see whether using some variable selection might improve the baseline routing performance. Note however that Buckley [8] finds that performance continues to improve with the number of terms in the routing query, even when thousands of terms are added.

7.4.3 Routing: Detailed look at Evaluation

While the evaluation results presented so far give the reader a basic idea about which retrieval methods are most successful, they hide a lot of detail about how one method performs better than another. In general, because of the number of experiments conducted and the volume of evaluation data collected it is not productive to present complete evaluation results for every experimental run. However, it is illuminating to examine the routing figures in more detail, since they provide a better picture of what characterizes the differences between experimental runs.

Figure 7.3 plots precision vs. recall for the routing experiment conducted above. Clearly, the results remain roughly the same at all levels of recall. Figure 7.4 plots precision and recall vs. the number of documents retrieved and is much more revealing. The major benefits of using more than 120 factors occur in the first few documents retrieved. If 10 or more documents will be retrieved, there is no advantage to keeping more than 120 factors. A similar pattern extends to higher numbers of documents retained. The recall plot reveals that after about 50 documents are retrieved, there is no advantage to keeping more than 80 factors. Finally, by the time 100 documents are retrieved the plot suggests that even 40 factors will be sufficient for retrieval. At 100 documents retrieved, a 40 factor LSI model is even better than the basic term model.

A more detailed look at the data captures the basic relationship between performance and the number of factors in the LSI model for the routing problem. If the user is only going to examine a few documents, it is best to use as many factors as possible. However, if many documents will be examined, there is no particular benefit to retaining large numbers of factors.

7.5 Routing and Classification

In the standard retrieval model, the system ranks documents according to their inner product similarity with respect to a query. The solutions to the routing problem presented above use the same method, although information from known relevant

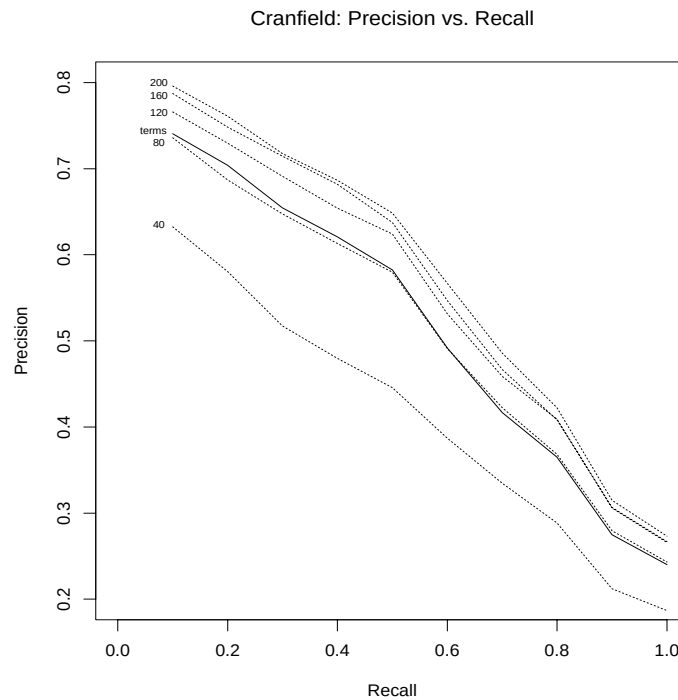


Figure 7.3: Cranfield: Precision vs. Recall

documents is incorporated into the query construction process. Ranking documents according to their inner product similarity to a query is equivalent to applying a one-dimensional linear projection to the document space. A well-constructed query will provide a nice separation between the relevant and non-relevant documents. In general, there is no a priori reason to believe that the inner product similarity to the query is the best linear projection of the documents. However, when no additional information is available, there is unlikely to be a better method for identifying relevant documents. However, the routing problem has far more information which can be exploited. In particular, we can make use of the distribution of the relevant documents.

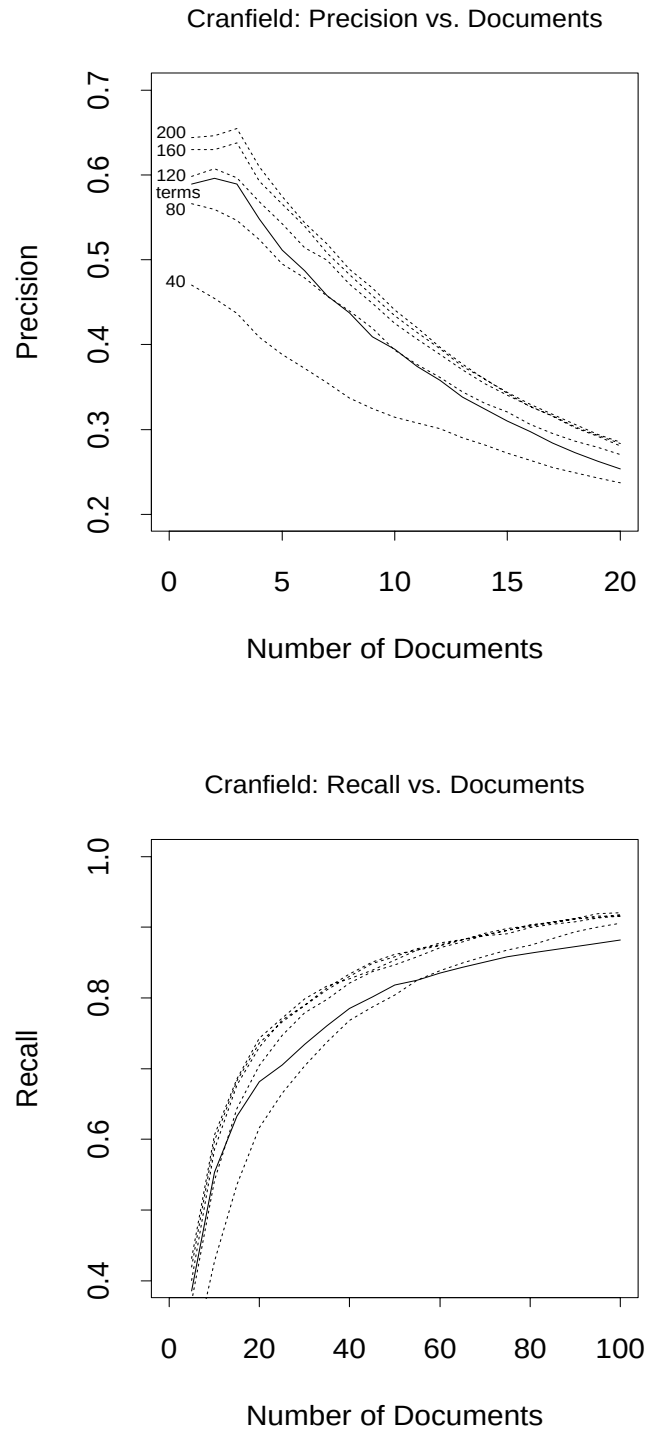


Figure 7.4: Cranfield: Precision and Recall vs. Documents Retrieved

7.5.1 Initial Experiment

Our next experiments apply the classification techniques developed in chapter 6 to the routing problem. Due to the limited number of relevant documents associated with each query, we must rely once again on cross-validation to obtain an unbiased estimate of performance. In the previous routing experiment, it was only necessary to apply cross-validation to the relevant documents since the classification rule was derived independent of the non-relevant documents. However, the classification techniques that will be applied here will all depend on the distribution of the non-relevant documents, so cross-validation of all observations must be attempted. Fortunately, for linear discriminant analysis, there exist updating formulae which will give the cross-validated distance of an observation from the group means without explicitly recomputing the mean vectors and the pooled covariance matrix [53].

A series of experiments attempted to improve performance by using linear discriminant analysis with documents as observations and different set of LSI variables as features. It quickly became apparent that LDA was significantly less effective than simply using the mean of the relevant documents. Clearly, it is not possible to obtain a good estimate of the covariance matrix even in the lower dimensional space provided by the LSI decomposition, as there are still 200 variables and only 1396 observations. Several variable reduction strategies were applied to test LDA on a subset of the features. The selection criteria included magnitude of the singular value and ability to separate the group means. However, none of the models fit using subsets of the variables provided better performance.

The performance figures obtained using the mean of the relevant documents indicate that all or almost all of the feature variables make some contribution to modelling document relevance. Therefore, it is not surprising the subset selection strategies are not effective. However, the full covariance matrix cannot be estimated accurately. Regularization could be attempted, but since we are already using cross-validation to measure performance, it would be extremely difficult to estimate the regularization parameters. Some alternative strategy must be adopted.

The LSI factors are derived to maximize the amount of variability explained over the entire collection. This representation is not necessarily optimal for characterizing

the region of the predictor space containing the majority of the relevant documents. It must be possible to parameterize the important features in the local region surrounding the query using a much more economical representation. In order to make this strategy work, we must compute a revised set of feature variables that is unique for each query and designed to capture structure in the local region surrounding the query.

7.5.2 A Local LSI Representation

Since the LSI representation is so effective for dimension reduction on a global level, it makes sense to apply it on the local level as well. In order to characterize the structure of the feature space, the singular value decomposition is applied to a matrix consisting only of the known relevant documents. Once again, only a small number of the most important factors are retained. This representation characterizes the important sources of variability which are present in the relevant documents. The non-relevant documents are folded in to this space using the usual strategy, as described in chapter 4.

The local representation could be computed directly from the original word space, but this was not found to be as effective, because the term matrix of relevant documents is too sparse to provide enough useful information. Instead, the global LSI representation of relevant documents is used as the basis for the local decomposition. It might not be clear immediately how capturing the structure of the relevant documents is useful for distinguishing them from the non-relevant documents, but a graphical representation of the reduced space provides an effective demonstration.

Figure 7.5 plots the document scores for two local LSI factors taken from the results for one query from the Cranfield collection. The coordinates in the plot are the inner product scores of the documents with respect to the first and second local LSI factors. These dimensions do not have any particular meaning for the non-relevant documents, so they tend to cluster around the origin. On the other hand, the relevant documents are more widely distributed in the document space, illustrating that the local factors are designed to capture their structure.

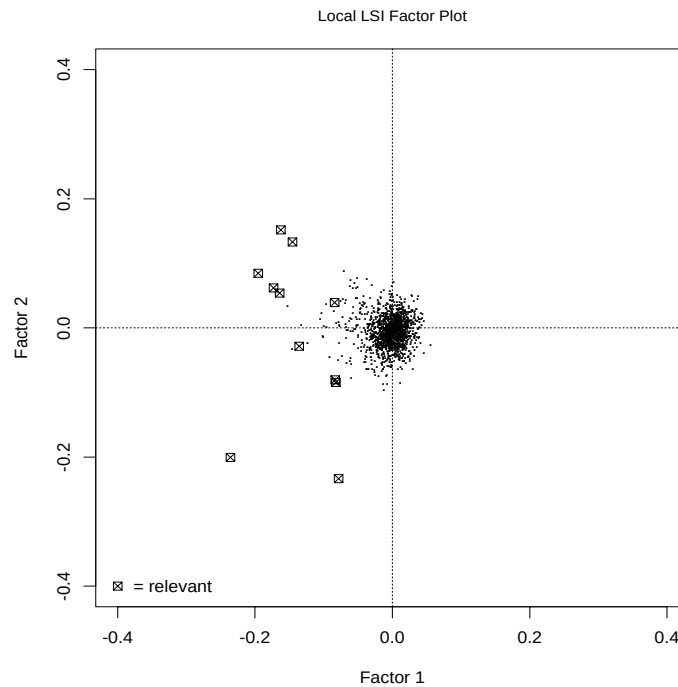


Figure 7.5: Cranfield: Local LSI Factor Plot

The local LSI method has an increased cost because a separate SVD be computed for each query. However, the SVD is only being applied to the relevant documents, which means that the matrix is far smaller than the one used in the initial LSI decomposition. In addition, since documents are being represented in LSI space, the matrix has at most a few hundred columns. For a matrix of this size, the first few singular vectors that are required can be computed extremely rapidly. However, the location of all documents in local space must also be computed.

We can now consider applying classification techniques in the local space. Since the relevant documents can appear anywhere in the space and there is no guarantee that they will be clustered on only one side of the origin, it is important to apply a

non-linear classification method in this example. The number of dimensions is now very small compared to the number of variables, and the covariance matrices of the two groups are dramatically different, so quadratic discriminant analysis is clearly appropriate. The next section presents the results of some retrieval experiments.

7.5.3 Experimental Results

In this set of experiments, the local LSI representation of the collection is computed as described above for each query in the collection. Quadratic discriminant analysis is used to obtain a score for each document indicating its relative distance from the relevant and non-relevant population. Cross-validation is used to obtain unbiased estimates of the distance. Note that the local LSI representation must be recomputed to obtain the cross-validated distance estimate for each relevant document. Only the first two factors are retained in the local representation. This number is not unreasonable considering that there are an average of only eight relevant documents per query.

Performance using the local representation is compared to our previous results for the routing problem:

	VSM	global LSI	local LSI
avg.prc	0.509	0.567	0.760

The combination of QDA and local LSI provides three times as much improvement as using global LSI alone, which clearly represents a dramatic improvement in performance. All methods are significantly different using both the Friedman test and the two-way ANOVA.

We repeat the experiment using three, four, and five local LSI factors with QDA. The results, broken down by the number of relevant documents in the query, are presented below:

num.rel	queries	global	local.2	local.3	local.4	local.5
3	29	0.568	0.363	NA	NA	NA
4	19	0.665	0.754	0.465	NA	NA
5	26	0.560	0.854	0.693	0.433	NA
6	28	0.485	0.853	0.795	0.642	0.463
7-8	36	0.631	0.877	0.906	0.843	0.693
9-10	29	0.606	0.852	0.908	0.928	0.894
11+	52	0.515	0.747	0.811	0.855	0.880

The local LSI decomposition cannot produce more dimensions than there are relevant documents. Since cross-validation removes one relevant document, there must be more relevant documents than local LSI factors to obtain a valid solution. In addition, the LSI factors are subject to a normalization constraint (the rows must be of unit length) so using a number of factors that is very close to the total number of relevant documents is not productive. Performance seems to peak at a number of dimensions roughly equal to half the number of relevant documents.

It seems that modelling local structure is the key to obtaining a good set of features for use in statistical classification. A representation based on the distribution of the relevant documents alone proves to be remarkably effective at separating relevant and non relevant documents. It may be helpful to include some nearby nonrelevant documents in the local decomposition, but that hypothesis still needs to be investigated. However, before we conclude that this method is likely to be universally successful, it is important to inject a cautionary note about the quality of the test collection.

7.6 Discussion

There are a number of reasons why the Cranfield collection may not be the ideal testing ground for information retrieval experiments. Some of them are described in chapter 5, where the problems associated with performing computer experiments using test collections are discussed. The collection has a relatively small number of documents and the documents are abstracts, which tend to be a paragraph or two at

most. The collection covers an extremely limited domain, aeronautical engineering, which is of a highly technical nature. This means that the queries and documents have large numbers of technical terms which have a relatively specific meaning and thus tend to be very useful for retrieval. In addition, it is well known that abstracts tend to have a higher than usual frequency of important content-bearing terms.

The 225 queries included with the collection have a total of 1838 relevant documents, with documents that are relevant to multiple queries being counted more than once. Of the 1397 documents in the collection 924 are relevant to at least one query, leaving 473 that are never relevant. This is an extraordinarily high signal to noise ratio and is almost certainly an unrealistic simulation of actual search conditions. Specifically, the documents will tend to cluster in groups that are relevant to one or more queries, and the feature space will not have a balanced distribution of non-relevant documents. These factors will make techniques (such as discriminant analysis) that are designed to measure similarity structure work particularly well in retrieval experiments.

Furthermore, there are a fair number of very short documents that are extremely similar to one another, having an important word or phrase in common. For example, there are author fields included in the documents, and multiple documents by the same author tend to be highly related and the author's name becomes an excellent discriminating variable. For the routing experiments, which use cross-validation, this causes particular problems, since cross-validation will obtain extremely favorable results if there are two very similar documents. The one remaining in the classification rule will make it very easy to determine that the other is relevant. In practice, it is extremely unlikely that there will be such a favorable distribution of documents. This situation causes particular problems because it biases the results much more in favor of classification methods that use the covariance of the relevant documents, like the local LSI and quadratic discriminant analysis, than other methods, such as those using the centroid of the relevant documents.

Therefore, it is important to try to extend the results that have been obtained here to a much larger collection that is less likely to suffer from the problems described above. This collection should cover a more general domain and contain longer, more

balanced documents. There should be enough relevant documents that it is possible to divide the collection into a training set and a test set so that cross-validation is longer necessary. Experiments on such a collection are conducted in the next chapter.

Chapter 8

Retrieval Experiments: Wall Street Journal

In 1992, a new experimental setting and test collection were developed in order to motivate work to improve the state of the art in information retrieval technology. The Text Retrieval Conference (TREC) brought together researchers from around the world to test the next generation of retrieval systems on a common test collection [34]. Known as the TREC collection, it now consists of over a million documents, from such sources as the Wall Street Journal, AP newswire, the Federal Register, as well as a collection of articles from computer journals. The TREC experiments are designed to accomplish two goals: (1) to provide a unified testing ground that will allow comparisons between a wide variety of different retrieval strategies and (2) to produce a test collection of realistic size so that experiments may be conducted on full text documents and the results should be more likely to generalize to an operational search environment.

The TREC collection is an ideal choice for our next set of retrieval experiments, since the issues that caused problems in experiments with the Cranfield collection; size, coverage, and number of relevant documents, will no longer be a significant concern. However, the full collection is so large that the computational costs of retrieval experiments will create new problems. Fortunately, it is possible to obtain the benefits of a large collection without using all of the TREC documents. Therefore,

we will restrict our attention to a subset of the full corpus, a collection of articles from the Wall Street Journal, covering the years 1987-1992.

Once again, the experiments presented here focus on the routing problem. For methods based on statistical classification, routing is the most tractable of the retrieval tasks, since a large amount of prior information (in the form of evaluated relevant documents) is available. Unless classification methods improve performance for the routing problem, there is little hope that they will work successfully for other techniques, such as query-based search or relevance feedback. The size of the new collection makes it much easier to evaluate performance for the routing problem.

This chapter presents a number of retrieval experiments, following the style of the previous chapter. It summarizes the important properties of Wall Street Journal (WSJ) documents, and discusses the problem with scaling traditional retrieval techniques to a collection that is several orders of magnitude larger than the ones used in other retrieval experiments. In particular, the LSI decomposition cannot be accurately computed for the full collection. It introduces the concept of screening, where the subset of the collection that is likely to contain the majority of the relevant documents is identified using a simple retrieval strategy, and more sophisticated models are applied to obtain a more accurate ranking of the remaining documents. Various statistical models are applied with limited success, as it is very difficult to improve upon the basic retrieval model for this collection.

8.1 The Wall Street Journal

The Wall Street Journal is a well-known newspaper whose articles primarily concentrate on financial topics. The collection included in TREC consists of approximately 180,000 documents and 250,000 unique terms after stemming and stop list removal, which makes it two orders of magnitude larger than the Cranfield collection. The average document length is roughly 300 indexed terms. While there are 150 queries available for the full text collection, we will select a sample of forty (from those labelled 51-100) for use in our experiments.

For the experiments, the collection is divided into a training set and a test set. The

training set consists of 100,000 documents from 1987-1989 and the test set consists of 80,000 documents from 1990-1992. The queries are considerably longer and more involved than the ones used with the Cranfield collection. For example, a typical query is:

Domain: International Economics

Topic: Rail Strikes

Description:

Document will predict or anticipate a rail strike or report an ongoing rail strike.

Narrative:

A relevant document will either report an impending rail strike, describing the conditions which may lead to a strike, or will provide an update on an ongoing strike. To be relevant, the document will identify the location of the strike or potential strike. For an impending strike, the document will report the status of negotiations, contract talks, etc. to enable an assessment of the probability of a strike. For an ongoing strike, the document will report the length of the strike to the current date and the status of negotiations or mediation.

Concept(s):

1. rail strike, picket, stoppage, lockout, walkout, wildcat
2. rail union, negotiator, railroad, federal conciliator, brotherhood
3. union proposal, talks, settlement, featherbedding, cost cutting
4. working without a contract, expired contract, cooling off period

The forty queries actually used in the retrieval experiments are chosen according to the following criteria. In order to be considered acceptable, a query should have

a significant number of relevant documents in both the training set and the test set. Queries with very few relevant documents in the test set will tend to have a much larger variability in their performance, since the evaluation score will depend heavily on the ranking of a small number of relevant documents. This is not a desirable property. Queries with very few relevant documents in the training set are not suitable for methods based on statistical classification. Removing them can be justified, since the number of relevant training documents is known in advance and other retrieval strategies will hold more promise. Since the primary goal of these experiments is to test classification methods, it makes sense to remove these queries.

Summary statistics giving the number of relevant documents for each query over the training and test sets appear below:

	Min	Median	Mean	Max
train	19	60	91	439
test	9	41	46	141

This represents a considerable improvement over the average of eight relevant documents per query for the Cranfield collection. Classification methods should be much more reliable when applied to training sets with 50 or 100 relevant documents.

8.2 Initial Experiments

8.2.1 Baseline from the VSM

The first experiment computes a baseline level of performance for the routing problem using the Vector Space Model. The retrieval model is applied in the same manner that is described in chapter 7, using the best term-weighting strategy found in those experiments.

$$\text{Query}(Q): q_i = \sqrt{(qtf)_i} * \log N/n_i^2 \quad , \quad \text{Document}(D_j): d_{ij} = \sqrt{\frac{(dtf)_{ij}}{\sum_j (dtf)_{ij}}}$$

Once again, the routing query is represented by the centroid of the original query and the relevant documents in the training set. The original query is given a weight

equal to five relevant documents and the individual terms are otherwise weighted in the same fashion as given above for documents. One can view the original query as a prior estimate of the word frequency parameters of the relevant documents.

The system computes the inner product similarity between the routing query and the documents in the test set, and documents are ranked in order of their similarity score. As a concession to the size of the collection, only the top 5000 documents are retained for purposes of evaluation. Performance will be evaluated in this chapter using three methods: average precision at 10% to 100% recall (prc.rec), average precision at 10 to 100 documents retrieved (avg.prc), and average recall at 100 to 1000 documents retrieved (avg.rec). The results for our sample of forty queries is given below.

	prc.rec	avg.prc	avg.rec
score	0.299	0.350	0.766

These scores will serve as the baseline results for future comparisons.

8.2.2 LSI on the WSJ

In the retrieval experiments conducted in the previous chapter, we found that Latent Semantic Indexing (LSI) produced a low dimensional representation of the Cranfield collection without decreasing retrieval performance. In fact, the reduced representation tended to produce slightly improved performance when applied to the routing problem. The next goal is to see whether LSI can provide the same level of performance on the new, much larger collection of articles from the Wall Street Journal. Once again, dimension reduction is necessary to transform the feature space so that it can be used with statistical classification algorithms.

On a collection with 250,000 terms (n) and 180,000 documents (k), it is a real challenge just to compute the singular value decomposition, even with a sparse matrix algorithm. Since the Lanczos method is $O(nk)$, it requires more than 100 billion operations to find the SVD for term-document matrix given above. Therefore, it is advisable to try to apply LSI to a smaller matrix of terms and documents. The first

LSI experiment on the Wall Street Journal computes the SVD on a symmetric term covariance matrix, using only 10,000 terms in the collection.

A term covariance matrix is selected for the following reasons. First, in order to make the computing requirement more manageable, we must select a subset of the original set of terms and documents. Second, selecting a subset of documents will cause the resulting representation of the collection to be biased towards capturing the structure contained in the selected documents. Third, it is known that some terms contain more information than others, so less information will be lost if the proper subset of terms is selected. It is not known a priori which documents are likely to be most informative.

The 10,000 selected terms occur in between 80 and 1000 documents, and represent the middle frequency terms in the collection. Low frequency terms have less information because they only capture the relationship between a small number of documents. High frequency terms have less information because they occur in such a large number of documents. If X is the weighted document by term matrix, let $\tilde{X}$ be the matrix obtaining by taking the columns of X that correspond to the sample of 10,000 terms that has been selected. Then the term covariance matrix is simply $\tilde{T} = \tilde{X}^\dagger \tilde{X}$. Applying the SVD to $\tilde{T}$ is equivalent to applying the spectral decomposition, since it is symmetric, and yields the solution:

$$\tilde{T} = U^\dagger \Lambda U$$

Each term is represented by a row of U and the dimension of the problem can be reduced by deleting columns of U corresponding to the smallest eigenvalues.

Documents are represented in the revised space by taking the weighted combination of the term vectors that they contain, using the weight function given in the previous section. Vectors for terms not initially selected are then constructed from the newly determined document vectors. While a lot of the structure in the original document-term matrix is lost in the new representation of terms and documents, it is hoped that enough is retained to capture the most important structure present in the full collection.

This technique is applied and the 100 most significant factors are retained from

the matrix decomposition. While computing and storing more factors is clearly desirable, obtaining enough storage space quickly becomes a problem. The 100 dimensional representation of the full collection requires more than 150MB of storage. Before allocating very large amounts of disk space, we should conduct a few retrieval experiments to test the quality of the current representation.

The experiment is conducted in the same fashion as the LSI experiments in chapter 7. A routing query is constructed using the centroid of the original query and the relevant documents in the training set. Similarity between the routing query and documents is computed as follows:

$$sim(Q, D_j) = \sum_i q_i * \lambda_i * d_{ij}$$

where q_i and d_{ij} are the weights for the i 'th revised LSI factor for the query and document j respectively, and λ_i is the i 'th diagonal element of Λ .

It quickly becomes apparent that model based on 100 revised LSI factors performs dramatically worse than the full term-based model. In hindsight, it is obvious that if an LSI model of the Cranfield collection, which contains only about 1400 documents, requires 100 LSI factors to perform as well as the term-based model, that there is no way that 180,000 documents from the Wall Street Journal can be properly modelled using only 100 factors. A proper model of the full collection might well require more than a thousand factors, and the storage costs associated with such a representation would be prodigious. Furthermore, it is difficult to imagine obtaining successful results from a classification algorithm that must analyze thousands of features. In our work on the Cranfield collection, we discovered that obtaining a local representation of the feature space in the neighborhood of the query is crucial for improving performance. The following section describes how to produce local LSI decompositions for the WSJ collection.

8.3 Query-Specific Screening of the Collection

In the last section, we described the problems associated with computing an LSI representation of a very large collection. A full description of the collection requires

retaining very large numbers of factors, few of which are going to be important for any particular query. Therefore, it is much more useful to try to capture structure on a local level, where the important information can be represented by a much smaller number of factors. The only drawback is that a separate representation must be computed for each query.

The following strategy is adopted to characterize the feature space in the neighborhood of the query and to obtain improved performance for the routing problem.

- 1) An initial screening algorithm ranks the documents in the training set in order of their similarity to the query. A subset of the highest ranking documents are selected. These documents are said to define the local training set of the given query.
- 2) Latent Semantic Indexing is applied to the documents in the local training set, and the dimension of the feature space is reduced by retaining only a small number of the most important factors.
- 3) A classification rule is developed for the local training set based on the relevant and non-relevant documents contained in that region.
- 4) The screening algorithm is applied to the full test set to determine which documents appear in the region defined by the local training set. The documents that satisfy the screening test become known as the local test set.
- 5) The classification rule is applied to the local test set and documents in this subset of the test collection are ranked in order of their estimated relevance. All other test documents (those that failed the screening test in (4)) are assumed to be not relevant.

The screening process is explained in more detail below.

The screening process is designed to identify documents that are obviously not relevant to the query and remove them from further consideration. It should use a method that can be computed reasonably efficiently over the entire document collection. The method selected does not need to accurately separate the relevant and

non-relevant documents immediately, however it should perform well at high recall, since it is important to collect as many relevant training and test documents in the local region as possible.

Our experiments rely on the Vector Space Model for screening the document set, using the term-weighting strategy described in chapter 7. This is identical to the baseline experiment presented in the previous section except that the document scores are computed over the training set as well as the test set. For each query, the 2000 highest ranking documents from the training set are selected to define the local region. The score of the 2000'th training document defines the boundary of the local region. The routing query is then applied to the test set and all documents with scores that are large enough to be inside the local boundary are selected to define the local test set.

It is difficult to determine the best way to select the size of the local region. It should be large enough to include as many relevant documents from the training set and test set as possible and large enough so that the local test set will contain significantly more documents than the user plans to examine. These criteria are necessary so that subsequent methods applied to the local region will have the potential to improve performance. Any relevant documents that don't make the cut for the local test set will never be discovered, no matter how good the local classification algorithms are. If the local test region contains fewer or only slightly more documents than the user plans to examine, then reordering these documents using a local classification method is basically pointless.

However, there are also several reasons why the local region should not be too large. Adding large numbers of non-relevant documents of marginal value will only increase the number of LSI factors needed to describe the local region without improving their quality, which will only degrade the performance of a classification algorithm. The local methods will be much more computationally intensive, and thus it is desirable to keep the number of observations small enough to ensure that the solution can be computed in a reasonable length of time.

The obvious solution would be to choose a fixed size for the local test set. Unfortunately, this is not a legal option under the constraints of the problem, because the

routing task assumes that the system sees the test documents one at a time and must decide whether to accept or reject a document without knowing the distribution of the full test set. Therefore, the only reasonable alternative is to choose a fixed document cut-off for the training set. The value of 2000 is selected because it should ensure that an average of 1600 documents make it into the local test set, assuming that the training and test sets are identically distributed. Samples of this size include the vast majority of the relevant documents from the training set, but are still tractable for classification problems and the singular value decomposition.

Of course, there is no guarantee that the documents from the test set will occur in the local region with the same frequency as those from the training set. For the Wall Street Journal, the training covers 1987-1989 while the test set covers 1990-1992. Issues that received a lot of coverage in the former period may no longer be important in the latter period and vice-versa. In fact, many of the issues covered by the queries will be transient in nature, so there are likely to be wild variations in the size of the local test set. This is an unfortunate but unavoidable problem, since it is generally difficult to predict the longevity of a particular issue. However, a careful user could take this factor into account when deciding on the size of the local region.

Furthermore, even if the the training and test set have proportionally similar numbers of documents in the local region, this does not guarantee that the document sets have the same distribution. Term space has hundreds of thousands of dimensions, which means that although both the local training documents and the local test documents are similar to the routing query, they need not be very similar to one another. However, no classification method will be successful if the training documents have a substantially different distribution than the test documents, so this issue does not dramatically affect the screening process.

8.3.1 The Screening Process

In this experiment, we apply the Vector Space Model using the routing query and screen out all documents that do not receive a higher similarity score than the training document of rank 2000 for each query. Since this is a simulation experiment,

the complete distribution of the test documents is known, and we can examine the relevance of the issues discussed above using data from the test set. In particular, the size of the test set, the fraction of relevant documents retained in the local region, and the correlation between coverage of the training set and coverage of the test set can all be examined.

The summary below compares the density of documents of the local region in the training set to that in the test set. Since the local region contains $\sim 2\%$ of the training set, and the test set has 80,000 documents, we would expect the test sets to contain roughly 1600 documents.

	Min	Mean	Median	Max	std.err
test	481	1455	1520	2258	361

For the 40 queries from the Wall Street Journal, there is a slightly lower density of documents in the local region of the test set on average, although on five occasions fewer than 1000 documents were recovered.

While obtaining test samples of consistent size is helpful, it is more important to retrieve as many relevant documents in the local region as possible. A summary for the training and test sets is given below:

	Min	Mean	Median	Max
train	77.8%	98.7%	100.0%	100.0%
test	44.3%	88.9%	94.2%	100.0%

It is not surprising that the local training set contains almost all the relevant documents, since the routing query is determined using the mean of the relevant documents in the training sets. However, there are a few queries which are very hard for the Vector Space Model, for which a number of relevant documents are outliers. Performance is considerably lower over the test set, but still recall is relatively high. For most queries, one can expect to pay a small penalty (5-10%) in terms of recall by using the screening algorithm, but for a few queries, that penalty can be as high as 50%.

It is also useful to know how well performance over the training set, which can be measured in advance, correlates with performance over the test set. Figure 8.1

presents a scatter plot of the percentage of relevant retrieved in the training set versus the test set. The correlation between these measures is .72, indicating that a relationship exists. From the picture, one can say that poor performance over the training set is a good indicator that relevant documents are being lost from the test set, but that good performance over the training set does not necessarily imply the contrary. However, if it finds all relevant training documents the screening process should also find the majority of the relevant test documents.

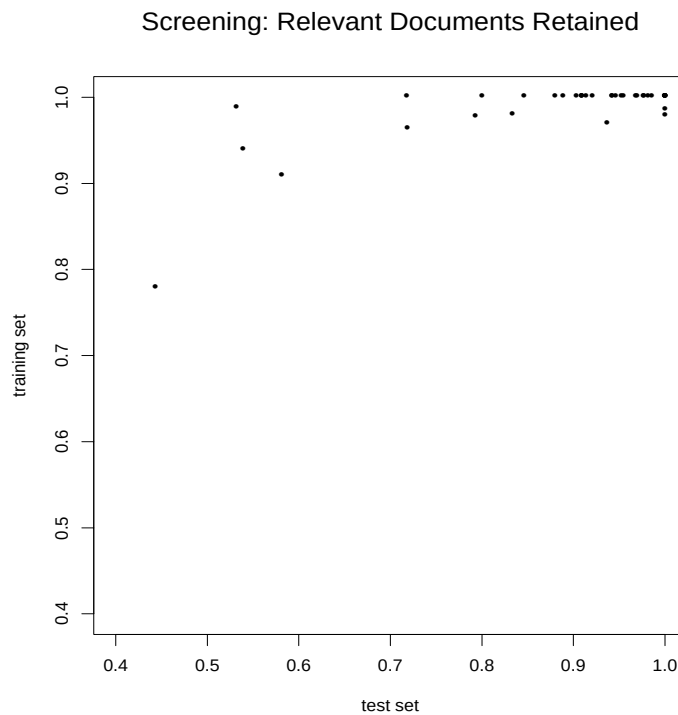


Figure 8.1: Relevant documents retained in local region

8.3.2 Local LSI

In the previous section, we determined that screening documents using the Vector Space Model can substantially reduce the scale of the document collection and in general leads to only a small loss in recall over the test set, although there are a few exceptions. Therefore, we can now consider applying more computationally intensive techniques in the local region, starting with Latent Semantic Indexing. Each query is characterized by a local training set of 2000 documents, a sample size for which the singular value decomposition can easily be computed using a sparse matrix algorithm.

We can make the problem more tractable by including only those terms that occur in the relevant documents, since only these terms have any discriminatory power for a classification rule. One can certainly argue that other terms may determine important structure in the local region, but there are sometimes as many as 200,000 unique terms in the local training set, so using all of them would remove the computational benefits of working over the local region. The relevant documents alone sometimes contain as many as 15,000 terms, so this sample is further reduced by restricting it to terms that occur in more than one relevant document. The distribution of the selected terms over the local training sets is then the following:

	Min	Median	Mean	Max	std.dev
train	983	2076	2636	6745	1480

The large standard deviation reflects the wide spread in the number of relevant training documents over different queries. A query with only 20 relevant documents will have far fewer eligible terms than a query with 400 relevant documents.

It should be mentioned that the local region has a much higher density of relevant documents than the collection as a whole, which can only help the performance of classification algorithms. The singular value decomposition is applied to the local training set and a representation of documents and terms is obtained based on local LSI factors. The local LSI representation is then computed for the documents in the test set using weighted combinations of the term vectors. Note that the documents in the local test set cannot be included in the SVD, since the classification rule must be constructed before the test documents are available.

The quality of the LSI representation can be measured by constructing the routing query using the mean of the relevant training document vectors as represented by their LSI factors. The similarity between the routing query and the test documents is computed in the same fashion described in chapter 7. Performance figures based on the number of LSI factors used in the model are given below:

	terms	40	80	120	160	200
prc.rec	0.299	0.321	0.336	0.347	0.346	0.346
avg.prc	0.350	0.371	0.383	0.392	0.389	0.391
avg.rec	0.766	0.803	0.808	0.812	0.813	0.812

Each of the LSI models is compared to the term baseline using the paired t-test and the Wilcoxon test. The LSI models with 80 or more factors are better than the baseline according to all three evaluation measures, and the LSI model with 40 factors is better than the baseline according to average recall.

Once again, Latent Semantic Indexing can be used to reduce the dimensionality of the problem and it actually improves retrieval performance by eliminating spurious relationships between terms. On average, the most important local structure can be captured using 100-120 LSI factors. Performance remains constant for 120-200 LSI factors. Screening and LSI have accurately captured the important relationships at a local level. It only remains to test the performance of statistical classification algorithms.

8.4 Classification Experiments

Classification techniques can be applied directly to LSI representation of the local training set. These methods should be more effective than they were for the Cranfield collection due to the much higher density of relevant documents. The following experiments will examine the effectiveness of discriminant analysis and logistic regression using the local LSI representation of documents.

8.4.1 LDA and Logistic Regression

We start by applying two basic linear classification methods to the data, linear discriminant analysis (LDA) and logistic regression. Despite screening and dimension reduction using LSI, there are still a relatively large number of predictor variables, so the problem will be a challenge for even the most basic methods. Each method will produce a linear classification rule, which can be used to induce a ranking of the test documents in order of their estimated likelihood of relevance. Performance is evaluated below using the average precision and 10 to 100 documents examined:

	40	80	120
LSI	0.371	0.383	0.392
LDA	0.381	0.399	0.404
logit	0.351	0.340	0.329

Hypothesis tests determine that the logistic model is significantly worse than LDA or basic LSI at all numbers of factors. There is no difference between LDA and LSI. In fact, if you exclude the first query where LDA scores 0.582 and LSI score 0.373, the two methods have basically the same average performance. This experiment provides no evidence that classification techniques can be used to improve performance.

There are two potential explanations for the disappointing performance of linear discriminant analysis. First, the covariance matrix may be roughly spherical, in which case there is not likely to be a better estimator than the centroid of the relevant documents. It should be noted that in the LSI representation, the components of the centroid of the non-relevant documents are all basically zero except for the first component. This is due to the fact that there are very few relevant documents and the other singular vectors are constrained to be orthogonal to the first singular vector, which assigns roughly equal weight to each document. Therefore, using the centroid of the relevant documents as the query is roughly equivalent to the linear discriminant rule using a spherical covariance matrix.

Second, it may be that 2000 observations is not enough to accurately estimate a covariance matrix for 40-120 feature variables. In this case, it is possible that regularization of the covariance matrix may improve performance. The experiments

in the next section will address this issue by applying regularized discriminant analysis to the problem. However, we first examine this issue by applying LDA to important subsets of the feature set to see whether better estimates of the covariance matrix could counteract the information that is lost by deleting factors. None of the models based on subsets of the feature variables (which are not detailed here) lead to any improvement in performance.

For the Cranfield collection, we were able to dramatically improve performance by applying quadratic discriminant analysis to a second matrix decomposition obtained using only the relevant documents. This alternative to variable selection is discussed in chapter 7 and will also be attempted on the WSJ text collection. Unfortunately, it does not improve performance on the new collection. There are important differences between the two problems that may explain its failure. On Cranfield, the local decomposition is applied to data consisting of all the documents in the collection. For the Wall Street Journal, a screening process has already been conducted, and the remaining non-relevant documents are all relatively close to the relevant region. Therefore, it is much harder to separate the non-relevant documents in this problem. As mentioned at the end of the previous chapter, the data from the Cranfield collection is unusually clean, in the sense that it has a much smaller population of documents that are not relevant to any query.

The model based on Logistic Regression performs significantly worse than the other two models. It is immediately clear that we are trying to fit too many parameters to the model. As the number of parameters increases, the predictive power over the test set decreases. Tests for the significance of individual parameters reveal that only a few variables can be said to differ from zero with any degree of certainty. Variable selection is a common strategy for the logistic model as well, and while it works slightly more successfully in this context than for LDA, the best model is still not quite as good as the baseline results obtained for LSI.

Many queries also have a problem with outliers. The outlying observations are generally relevant documents that are assigned a very low probability of relevance (often $< .05$). Removing these observations improves performance slightly, but the difference is only enough to bring the performance of the method roughly up to par

when compared to the baseline LSI model. The best models also use variable selection as described above. From these experiments, we conclude that Logistic Regression is not well suited for the routing problem, due to the large number of feature variables that must be modelled.

8.4.2 Regularized Discriminant Analysis

The results of the previous experiment suggest that linear discriminant analysis is not the ideal method for solving the routing problem, since it cannot outperform a classification rule based on simply using the centroid of the relevant documents. There are several factors which may explain why LDA is not more successful. First, the estimate of the pooled covariance matrix may be poor due to the large number of feature variables. Second, LDA works best when the covariance matrices of the two classes are roughly equal, which may be a badly flawed assumption. Both of these problems can be mitigated by using regularized discriminant analysis, as described in chapter 6.

In RDA, the estimate of the within-group covariance matrix is biased using two regularization parameters [26]:

$$\hat{\Sigma}_i(\lambda, \gamma) = (1 - \gamma)\hat{\Sigma}_i(\lambda) + \frac{\gamma}{p}\text{tr}(\hat{\Sigma}_i(\lambda))I_p$$

$$\hat{\Sigma}_i(\lambda) = \frac{(1 - \lambda)(n_i - 1)S_i + \lambda(n - 2)S}{(1 - \lambda)(n_i - 1) + \lambda(n - 2)}$$

The regularization parameter λ measures the relative importance of the pooled and unpooled estimates of the covariance matrix, with $\lambda = 1$ corresponding to linear discriminant analysis. The parameter γ shrinks the estimate of the covariance matrix towards the identity, which adjusts for potential bias in the eigenvalues of the estimated covariance matrix. For more details, see chapter 6.

In the first experiment, we measure the impact of changes in the regularization parameters on performance for the routing problem. Friedman [26] measures performance on a 5-by-5 grid of values for λ and γ with each parameter ranging from zero to one. In general, Σ_r cannot be estimated, since the number of variables usually

exceeds the number of relevant documents. Therefore, we will require that $\lambda > 0$ and measure IR performance at the following levels:

$$\lambda = [0.2, 0.4, 0.6, 0.8, 1.0] \quad \gamma = [0.0, 0.25, 0.50, 0.75, 1.0]$$

Since not even the simplest model using the centroid of the relevant documents works better with more than 120 variables, we will limit the feature set to the first 120 LSI factors.

The tables below present the results of running regularized discriminant analysis using fixed values of the regularization parameters for all queries, for average precision:

$\lambda \backslash \gamma$	0.0	0.25	0.5	0.75	1.0
0.2	0.172	0.135	0.116	0.103	0.093
0.4	0.259	0.213	0.192	0.175	0.162
0.6	0.340	0.304	0.289	0.275	0.260
0.8	0.387	0.371	0.361	0.354	0.346
1.0	0.404	0.387	0.382	0.376	0.371

and average recall:

$\lambda \backslash \gamma$	0.0	0.25	0.5	0.75	1.0
0.2	0.500	0.466	0.454	0.442	0.430
0.4	0.588	0.555	0.542	0.531	0.524
0.6	0.674	0.655	0.647	0.641	0.637
0.8	0.747	0.752	0.750	0.754	0.754
1.0	0.789	0.800	0.803	0.804	0.802

On average, regularization away from LDA reduces performance when up to 100 documents are examined. The average precision drops quickly as the weight of Σ_r increases and drops more gradually as the covariance matrix is regularized towards the identity. When the number of documents examined increases to between 100 and 1000, the pattern is the same for regularization away from the pooled covariance matrix while regularization towards the identity makes no difference when $\lambda = 1$.

From these tables, it appears that there is no obvious improvement in performance from regularization away from linear discriminant analysis. However, this conclusion

applies only to the average results, and performance on a query by query basis may differ. Since the final goal is to select the regularization parameters using the training data, it is important to consider the potential improvement which could be obtained by selecting a separate amount of regularization for each query.

The entries in the tables below show the number of queries that attain their maximum score at the given setting of the regularization parameters. The first table measures average precision and the second table measures average recall.

$\lambda \backslash \gamma$	0.0	0.25	0.5	0.75	1.0
0.2	0	0	0	0	0
0.4	0	0	0	0	0
0.6	1	0	0	0	1
0.8	6	1	2	2	1
1.0	14	2	2	3	5

For roughly a third of the queries LDA provides the optimal performance, however the majority of the queries can benefit from some regularization away from that solution.

$\lambda \backslash \gamma$	0.0	0.25	0.5	0.75	1.0
0.2	0	0	0	0	0
0.4	0	0	0	0	0
0.6	0	0	0	0	0
0.8	2	2	0	3	4
1.0	10	8	1	2	8

More queries benefit from regularization towards the identity if large numbers of documents (more than 100) will be examined. The final table shows the performance levels that could be attained if the optimal value of the regularization parameters could be selected for each query.

	idf	LSI	LDA	RDA
avg.prc	0.350	0.392	0.404	0.422
avg.rec	0.766	0.812	0.789	0.817

While it might seem that these results indicate that RDA has some potential to improve performance, we must remember that the score for RDA is the theoretical maximum and will not be attained in practice because of the error associated with estimating the regularization parameters. The optimal values for RDA are distressingly small when one considers that they are maximized over 25 potential parameter settings.

The small difference between optimal RDA and LDA shown above appears to contradict the previous tables which suggest that the optimal settings for the regularization parameters vary widely from query to query. A closer look at the data reveals that for λ close to one performance is relatively constant over the entire range of γ . The optimal values are often greater than their neighbors by only one or two tenths of a percent.

The final experiments test to see how much of the observed benefits of choosing the optimal regularization parameters can be achieved in practice. Friedman [26] suggests estimating the regularization parameters by using cross-validation over the training set to select the grid point with the lowest misclassification rate. Unfortunately, this computation is not feasible for our training sets with 2000 observations and 200 variables. Therefore, we must find an alternative strategy.

We begin by splitting the training set into two parts, a smaller initial training set consisting of two thirds of the data and a validation set consisting of one third of the data. The initial training set is used to estimate the model parameters and the validation set is used to estimate the regularization parameters. The resulting model is then applied to the test set. Rather than the misclassification rate, average precision at 10 through 100 documents examined is used as the criterion to select the optimal values of the regularization parameters, since this is the evaluation criterion.

Unfortunately, this approach leads to worse performance than the model using LDA. This is probably due to the fact that the model parameters are being estimated using fewer data points, which results in a corresponding increase in the variance of their estimators. To get around this problem, we modify the strategy slightly. Once the regularization parameters are estimated, the validation set is merged back into the training set and the mean vectors and covariance matrices are re-estimated

using all the training data. However, this approach also fails. A closer look at the results reveals that the values of regularization parameters are biased towards too much regularization. This occurs because the amount of regularization depends to some extent on the ratio of the number of observations to the number of variables. When the regularization parameters are estimated on a smaller training set, they are not unbiased estimates of the parameters for a larger training set. While this effect is small, it is enough to wipe out any gains due to regularization.

As a final solution, we adopt the strategy of N-fold cross-validation, which is used for classification and regression trees [7]. This method partitions the training set into N randomly selected equally sized samples. The samples are removed one at a time and the remaining data is used to estimate the model parameters while the sample is used to evaluate performance. The performance figures are combined over all samples and the maximum score determines the optimal setting of the regularization parameters. For both 5-fold and 10-fold cross-validation, the final score is 0.407, which represents an insignificant improvement over using LDA. In fact, the setting selected for the regularization parameters is equivalent to LDA for 70-80% of the queries.

8.5 Discussion

The experiments presented in this chapter indicate that classification based on discriminant analysis does not improve performance when compared to a ranking of documents based on their inner product similarity score with respect to the query. As described previously, there are two possible explanations for these results. Either there is no benefit to looking at the distribution of the covariance matrix for the two classes, or the covariance matrices are poorly estimated. Since regularization did not substantially improve performance, it is likely that the former is true. There are additional reasons to favor this hypothesis.

Since the LSI factors are orthogonal, the covariance matrix of the full data set is diagonal. The vast majority of the represented documents are not relevant, so the covariance matrix of the non-relevant class and the pooled covariance matrix are also

roughly diagonal. Therefore, scaling by the covariance matrix is roughly equivalent to scaling each variable by its variance. The results from the previous section indicate that regularization towards the identity does little to change performance. Therefore, there is no reason to rescale distance estimates using the covariance matrix.

It is well known that word similarity statistics provide only an approximate representation of the concept of relevance. It appears likely that the information available is just not of high enough quality to benefit from a model which estimates a very large number of parameters. Some relevant documents will not share a lot of vocabulary with their fellow class members. The ranking of these documents will change very little, regardless of the classification technique. The relevant documents that do share a lot of vocabulary are often easy to identify using simple strategies. In realistic retrieval situations, as characterized by the documents from this collection, we are forced to conclude that classification based on discriminant analysis does not significantly improve performance for the routing problem.

Chapter 9

Conclusion

It has been recognized that effective information retrieval systems are vital for storing and managing the vast electronic databases that have been developed over the past few decades. The core components of IR systems are the search strategies used to identify documents relevant to the user's query. In order to successfully implement search methods, we must look for solutions to the basic problems in information retrieval.

The information retrieval problem can be expressed in many forms, ranging from a general desire to summarize the contents of a collection to the goal of seeking the answer to very specific questions. The work presented in this thesis addresses the classical IR problem, which is defined as finding all the documents in the collection that refer to the topic expressed by the user's query. This problem can be broken up into three categories, depending on the amount of information given to the system.

In the adhoc search problem, the system must rely solely upon a relatively short text-based query to define the relevant topic. The system analyzes the query and then ranks the documents in the collection according to their estimated relevance to the query. In relevance feedback, the basic search process goes through several iterations. At each iteration, the user examines one or more of the highest ranking documents, determines whether or not they are relevant, and submits this information to the system. The system uses the relevance data to obtain an improved ranking of the remaining documents in the collection. For the routing problem, the system

begins with a relatively large sample of evaluated documents and must estimate the relevance of newly arriving documents.

Of these problems, the routing task is best suited to statistical methods because a lot of data in the form of evaluated documents are available to estimate a model of the relevant topic. In fact, routing is merely a special case of the basic classification problem, where the goal is to predict which documents are most likely to be relevant. The IR problem is challenging for classification techniques due to the size of the data sets and the difficulties associated with finding a reasonable set of feature variables.

9.1 The Vector Space Model

The standard approach to the IR problem is to treat documents and the query as vectors of term frequencies and measure the similarity between a document and the query by the inner product of the corresponding term vectors. It is based on the assumption that a document is related to the query if the two have a lot of vocabulary in common, and this model works relatively well in practice. A large fraction of the most successful experimental IR systems rely on some variant of this basic model.

The solution to the Vector Space Model can be computed efficiently even over very large text collections, using the property that most words occur in only a few documents. Therefore, similarity scores need only be computed for documents that have terms in common with the query, and much of this information can be calculated in advance and stored in the inverted index. Performance can be optimized by the proper selection of term weights to reflect the fact that some terms are more important than others and that similarity scores must be normalized to account for document length.

The Vector Space Model can use any form of free text for the query, ranging from a few sentences written by the user to describe the topic of interest to the full text of a relevant document. This property allows the development of automatic relevance feedback strategies, where the user labels a set of documents according to their relevance and the system automatically incorporates text of the relevant documents into a revised query. In relevance feedback, these documents are labelled sequentially, as

the system computes updated rankings. For the routing problem, a large sample of labelled documents are available in advance, and they can be incorporated into the initial query.

9.2 Dimension Reduction and Classification

The Vector Space Model suffers from a number of weaknesses that suggest that it should be possible to produce an improved model. In particular, words are a relatively poor choice of features. Many words have the same meaning and some words have many different meanings. The average term occurs in a relatively small number of documents and thus carries a limited amount of information. The Vector Space Model is based on an assumption of term independence which is often violated over real collections. Many terms that relate to same topic will tend to occur together and common phrases have a very high degree of dependence. In order to improve performance, we must develop a better set of feature variables.

The first step to applying classification methods in information retrieval is to find a means of dimension reduction. The average collection contains thousands of terms, each of which is of only limited value. The most promising strategy is known as Latent Semantic Indexing and involves applying a singular value decomposition to the term-document matrix. Terms and documents are re-expressed as linear combinations of orthogonal indexing variables. Latent Semantic Indexing captures the vital structure in a document collection using only a few hundred singular vectors.

The IR problem has a number of unique features that make it difficult to solve using classification methods. Data sets are huge, consisting of thousands of documents and hundreds of features, even after dimension reduction. Relevant documents are usually relatively rare, making up a very small fraction of the document collection. The most successful classification methods are based on discriminant analysis, which models the population of relevant and non-relevant documents using the multivariate normal distribution.

9.3 Evaluation and Retrieval Experiments

The best way to test the performance of different retrieval strategies is using a computer simulation experiment. In this setting, a sample of queries is available and the documents which are relevant to each query have already been manually identified. The performance of each automatic system can then be compared to a known standard of optimal performance. Systems are rated according to their ability to rank the relevant documents higher than the documents which are not relevant. While one can make a number of arguments about how and why this test setting does not reflect reality, no better methods for evaluating performance have been developed.

Our experimental work begins on the Cranfield corpus, a small collection of documents on a technical subject. We develop baseline values for the Vector Space Model by optimizing over a number of different term-weighting strategies. When Latent Semantic Indexing is applied to the collection, we discover that performance actually improves using a representation consisting of roughly 200 factors. There are very few relevant documents per query in Cranfield so classification methods have little success on the routing problem even in the reduced space. However, when each query is modelled separately by a set of features that characterize the local space around the relevant documents, we obtain a dramatic improvement in performance. Unfortunately, the Cranfield collection has an unusually favorable structure for experiments based on cross-validation (which is necessary to obtain unbiased performance estimates for the routing problem on this collection) which indicates that the performance gains may not carry over to other collections.

Subsequent experiments are carried out on a very large selection of articles from the Wall Street Journal. This collection consists of almost 200,000 documents, so its size poses new challenges for classification algorithms. Storing and computing an LSI solution for the full collection proves to be a daunting task, and the global LSI factors simply do not capture enough of the structure in the region of the query to produce adequate performance in retrieval experiments. Rather, we must develop a local LSI representation for each query by applying the matrix decomposition to a sample of documents in the neighborhood of the routing query. This method proves

to be extremely successful, and the reduced LSI representation of local space once again leads to improved retrieval performance. However, linear discriminant analysis does not further improve performance. Regularization is used to try and obtain better estimates of the pooled covariance matrix, but the final results suggest that there is little benefit to regularization away from LDA, and that the optimal regularization parameters are difficult to estimate accurately.

9.4 Extensions of this work

The experiments conducted in chapter 8 demonstrate that it is difficult to improve performance for the routing problem using classification techniques based on discriminant analysis. However, this does not mean that we should give up on this direction of research. It seems that LDA fails largely because the pooled covariance matrix is almost diagonal. Therefore, LDA does little more than rescale the variables, and the scale of the variables does not make much difference in performance for this problem. There are several ways that we can address this issue.

It may be possible to select a better representation of the local region. In particular, our experiments did not study the impact of changing the number of documents used to define the local neighborhood. We have recognized that many relevant documents score poorly because they have very little vocabulary in common with the relevant topic, therefore no search method based on word similarity will have much chance to recover them. These outliers may well hurt the performance of the classification rule. If the local region shrinks to exclude these documents, classification techniques may have more success.

Furthermore, a smaller neighborhood will have a higher density of relevant documents, which should also help improve classification techniques, because a higher proportion of the LSI factors will be used to measure the structure of the relevant documents. The pooled covariance matrix will have richer structure because the distribution of the relevant documents can be measured in more detail. In previous experiments, when non-relevant documents outnumbered relevant documents by twenty to one, most of the LSI factors must measure the variability in the non-relevant

documents. These factors do not help much for classification and using them only adds to the variability of the model.

The retrieval experiments in chapter 8 were too ambitious, in the sense that they tried to capture all the relevant documents in the local region. By sacrificing recall in order to increase the density of relevant documents in the local neighborhood, classification methods may well be much more successful. That is the current direction of research being pursued by the author. However, it may also be possible to improve performance using alternative strategies. In particular, we could work to improve the query and document representations.

The models studied in this thesis represent queries by vectors of term frequencies. As described in chapter 4, words often do not make good features, and it may be helpful to supplement them with phrases or word clusters. For example, one TREC query is searching for “leveraged buy-outs”. All of these words occur commonly in other contexts, so identifying the entire phrase is crucial to obtain good performance. The experiments presented here assume that LSI is a surrogate for phrase construction. It may be beneficial to identify important phrases in the query before conducting an LSI decomposition. While an improved set of features will help all retrieval methods, it should be particularly helpful for more complex models based on statistical classification.

The Vector Space Model assumes that the content of a document is generally limited to a single topic. For the longer full-text documents contained in the Wall Street Journal, this may not be the case. As described in chapter 3, a lot of recent research has been devoted to partitioning documents into passages and using measurements based on passage similarity to improve retrieval performance. An improved representation of documents would also make classification algorithms more effective. Basically, any procedure which can improve the quality of the data makes it easier to estimate more complex models, which should make classification techniques more effective when compared to the basic model.

Until we have more success with the routing task, there is little hope in trying to applying classification techniques to other IR problems, such as relevance feedback, where even less information is available to the system. Relevance feedback is an even

more challenging problem because the system receives a biased training sample. Since the final goal is to obtain as many relevant document as possible, the user evaluates only those documents most likely to be relevant. Therefore, classification methods trained on this sample cannot be expected to generalize to the remaining less-relevant documents in the collection. Therefore, improving on the current methods of relevance feedback will require an alternative approach (for example [48]).

From the work presented in this thesis, it is clear that dimension reduction using Latent Semantic Indexing can be accomplished with no loss in retrieval performance, even for very large collections. However, the LSI representation must be computed separately in the local region of each query. Unfortunately, the issue of whether classification algorithms can further improve performance still remains in doubt. It appears that word features simply contain too much noise to be accurately modelled with sophisticated classification algorithms. However, our experiments worked on samples with a very small percentage of relevant documents. There is still some hope that by increasing the ratio of relevant to non-relevant documents, we can develop classification methods that will work better than the basic model for the routing problem. If this tactic fails, we must improve the quality of the data by finding better representations of queries and documents before we can expect to successfully apply classification algorithms.

Bibliography

- [1] The academic american encyclopedia. Grolier Electronic Publishing, 1990.
- [2] IJsbrand Jan Aalbersberg. Incremental relevance feedback. In *Proc. 15th Int'l Conference on R&D in IR (SIGIR)*, pages 11–22, 1992.
- [3] T.W. Anderson. *An Introduction to Multivariate Statistical Analysis*, chapter 11. Wiley, 2nd edition, 1984.
- [4] C. Apte, F. Damerau, and S.M. Weiss. Towards language independent automated learning of text categorization models. In *Proc. 17th Int'l Conference on R&D in IR (SIGIR)*, pages 23–30, 1994.
- [5] K. Bennett and O.L. Mangasarian. Robust linear programming discrimination of two linearly inseperable sets. Technical Report 1054, University of Wisconsin, 1991.
- [6] G.E.P. Box, W.G. Hunter, and J.S. Hunter. *Statistics for Experimenters*, page 144. John Wiley and Sons, 1978.
- [7] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth and Brooks, 1984.
- [8] C. Buckley, G. Salton, and J. Allen. 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.

- [9] Chris Buckley, James Allan, and Gerard Salton. Automatic routing and ad-hoc retrieval using smart: Trec 2. In *Proceedings of the 2nd Text Retrieval Conference (TREC-2)*, pages 45–55, 1994.
- [10] T. Cacoullos and I. Olkin. On the bias of functions of characteristics roots. *Biometrika*, 52(1 and 2):87–94, 1965.
- [11] James P. Callan. Passage-level evidence in document retrieval. In *Proc. 17th Int'l Conference on R&D in IR (SIGIR)*, pages 302–310, 1994.
- [12] C.W. Cleverdon, J. Mills, and E.M. Keen. *Factors Determining the Performance of Indexing System*. Cranfield, England: Aslib Cranfield Research Project, 1966.
- [13] W.J. Conover. *Practical Nonparametric Statistics*. John Wiley and Sons, 2nd edition, 1980.
- [14] W.S. Cooper. Expected search length: A single measure of retrieval effectiveness based on weak ordering action of retrieval systems. *Journal of the American Society for Information Science*, 19:30–41, 1968.
- [15] Carolyn J. Crouch. A cluster based approach to thesaurus construction. In *Proc. 11th Int'l Conference on R&D in IR (SIGIR)*, pages 309–320, 1988.
- [16] D. Cutting, J. Pedersen, and P.K. Halvorsen. An object-oriented architecture for text retrieval. In *Proc. of Intelligent Multimedia Retrieval Systems and Management Conference (RIAO)*, 1991.
- [17] Doug Cutting and Jan Pedersen. Optimizations for dynamic inverted index maintenance. In *Proc. 13th Int'l Conference on R&D in IR (SIGIR)*, pages 405–411, 1990.
- [18] D.R. Cutting, J. Pedersen, D. Karger, and J.W. Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In *Proc. 15th Int'l Conference on R&D in IR (SIGIR)*, pages 318–329, 1992.

- [19] 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.
- [20] Susan T. Dumais. Latent semantic indexing (lsi) and trec-2. In *The Second Text REtrieval Conference (TREC-2)*, pages 105–115, 1993.
- [21] Michael Berry et. al. *SVDPACKC (Version 1.0) User's Guide*. Department of Computer Science, University of Tennessee, 1993.
- [22] Joel L. Fagan. *Experiments in Automatic Phrase Indexing for Document Retrieval: A Comparison of Syntactic and Non-Syntactic Methods*. PhD thesis, Department of Computer Science, Cornell University, 1987.
- [23] W.B. Frakes and R. Baeza-Yates, editors. *Information Retrieval: Data Structures and Algorithms*. Prentice-Hall, 1992.
- [24] H.P. Frei and P. Schauble. Determining the effectiveness of retrieval algorithms. *Information Processing and Management*, 27(2/3):153–164, 1991.
- [25] J. Friedman, J. Bentley, and R. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software*, 3(3):209–226, 1977.
- [26] Jerome H. Friedman. Regularized discriminant analysis. *Journal of the American Statistical Association*, 84(405):165–175, 1989.
- [27] Norbert Fuhr. A probabilistic relational model for the integration of ir and databases. In *Proc. 16th Int'l Conference on R&D in IR (SIGIR)*, pages 309–317, 1993.
- [28] William Gale, Kenneth Church, and David Yarowsky. Discrimination decisions for 100,000 dimensional spaces. Technical Report 103, AT&T Bell Laboratories, 1992.

- [29] R. Gnanadesikan. *Statistical Data Analysis of Multivariate Observations*, chapter 4, pages 82–96. John Wiley and Sons, 1977.
- [30] Gene Golub and Charles Van Loan. *Matrix Computations*, chapter 5. Johns Hopkins Press, 2nd edition, 1989.
- [31] Gregory Grefenstette. Use of syntactic context to produce term association lists for information retrieval. In *Proc. 15th Int'l Conference on R&D in IR (SIGIR)*, pages 89–97, 1992.
- [32] D. Harman. How effective is suffixing? *Journal of the American Society for Information Science*, 42(1):7–15, 1991.
- [33] Donna Harman. Relevance feedback revisited. In *Proc. 15th Int'l Conference on R&D in IR (SIGIR)*, pages 1–10, 1992.
- [34] Donna Harman, editor. *Proceedings of the 1st Text Retrieval Conference (TREC-1)*, 1993.
- [35] Donna Harman, editor. *Proceedings of the 2nd Text Retrieval Conference (TREC-2)*, 1994.
- [36] Marti Hearst and Christian Plaunt. Subtopic structuring for full-length document access. In *Proc. 16th Int'l Conference on R&D in IR (SIGIR)*, pages 59–68, 1993.
- [37] Marti A. Hearst. Texttiling: A quantitative approach to discourse segmentation. Technical Report 93/24, University of California, Berkeley, 1993.
- [38] William Hersh. Relevance and retrieval evaluation: Perspectives from medicine. *Journal of the American Society for Information Science*, 45(3):201–206, 1994.
- [39] M. Hollander and D. Wolfe. *Nonparametric Statistical Methods*, chapter 3, 7, pages 27–49, 138–184. John Wiley and Sons, 1973.
- [40] David Hull. Using statistical testing in the evaluation of retrieval performance. In *Proc. of the 16th ACM/SIGIR Conference*, pages 329–338, 1993.

- [41] R.L. Iman and J.M. Davenport. Approximations of the critical region of the friedman statistic. *Communications in Statistics*, A9:571–xxx, 1980.
- [42] Karen Sparck Jones. Collection properties influencing automatic term classification performance. *Information Storage and Retrieval*, 9:499–513, 1973.
- [43] Guy L. Steele Jr. *Common Lisp: the Language*. Digital Press, 2nd edition, 1990.
- [44] E. Michael Keen. Presenting results of experimental retrieval comparisons. *Information Processing and Management*, 28(4):491–502, 1992.
- [45] Julian Kupiec. Murax: A robust linguistic approach for question answering. In *Proc. 16th Int'l Conference on R&D in IR (SIGIR)*, pages 181–190, 1993.
- [46] F.W. Lancaster. *Information Retrieval Systems: Characteristics, Testing and Evaluation*. Wiley, 1968.
- [47] E.L. Lehmann. *Testing Statistical Hypotheses*, chapter 3.1, pages 68–72. John Wiley and Sons, 2nd edition, 1986.
- [48] David Lewis and William Gale. A sequential algorithm for training text classifiers. In *Proc. 17th Int'l Conference on R&D in IR (SIGIR)*, pages 3–12, 1994.
- [49] David Lewis and Marc Ringuette. A comparison of two learning algorithms for text categorization. In *Symposium on Document Analysis and Information Retrieval*. University of Nevada, Las Vegas, 1994.
- [50] David D. Lewis and W. Bruce Croft. Term clustering of syntactic phrases. In *Proc. 13th Int'l Conference on R&D in IR (SIGIR)*, pages 385–404, 1990.
- [51] Harold R. Lindman. *Analysis of Variance in Complex Experimental Designs*. W.H. Freeman and Co., 1974.
- [52] P. McCullagh and J.A. Nelder. *Generalized Linear Models*, chapter 4, pages 101–123. Chapman and Hall, 2nd edition, 1989.

- [53] Geoffrey J. McLachlan. *Discriminant Analysis and Statistical Pattern Recognition*. John Wiley and Sons, 1992.
- [54] Jack Minker, Gerald Wilson, and Barbara Zimmerman. An evaluation of query expansion by the addition of clustered terms for a document retrieval system. *Information Storage and Retrieval*, 8:329–348, 1972.
- [55] Frederick Mosteller and David Wallace. *Applied Bayesian and Classical Inference: The Case of the Federalist Papers*. Springer-Verlag, 1964.
- [56] B.A. Murtagh and M.A. Saunders. Minos 5.4 user's guide. Technical Report SOL 83-20R, Stanford University, 1993.
- [57] J. Neter, W. Wasserman, and M. Kutner. *Applied Linear Statistical Models*. R.D. Irwin, 2nd edition, 1985.
- [58] Art Owen. Class Notes - Statistics 348, Stanford University.
- [59] T.K. Park. Towards a theory of user-based relevance: a call for a new paradigm of inquiry. *Journal of the American Society for Information Science*, 45(3):135–141, 1994.
- [60] Michael Persin. Document filtering for fast ranking. In *Proc. 17th Int'l Conference on R&D in IR (SIGIR)*, pages 339–348, 1994.
- [61] Yonggang Qiu and H.P. Frei. Concept based query expansion. In *Proc. 16th Int'l Conference on R&D in IR (SIGIR)*, pages 160–169, 1993.
- [62] V. Raghavan, P. Bollmann, and G. Jung. Retrieval system evaluation using recall and precision. In *Proc. 12th Int'l Conference on R&D in IR (SIGIR)*, pages 59–68, 1989.
- [63] J.J. Rocchio. Relevance feedback in information retrieval. In Gerard Salton, editor, *The SMART retrieval system: Experiments in Automatic Document Processing*, pages 313–323. Prentice-Hall, 1971.

- [64] G. Salton, E. Fox, and H. Wu. Extended boolean information retrieval. *Communication of the ACM*, 26(11):1022–1036, 1983.
- [65] G. Salton and M.E. Lesk. Computer evaluation of indexing and text processing. *Journal of the Association for Computing Machinery*, 15(1):8–36, 1968.
- [66] Gerard Salton, editor. *The SMART retrieval system*. Prentice-Hall, 1971.
- [67] Gerard Salton. *Automatic Text Processing*. Addison-Wesley, 1989.
- [68] Gerard Salton. The state of retrieval system evaluation. *Information Processing and Management*, 28(4):441–449, 1992.
- [69] Gerard Salton and Chris Buckley. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 41(4):288–296, 1990.
- [70] Mark Sanderson. Word sense disambiguation and information retrieval. In *Proc. 17th Int'l Conference on R&D in IR (SIGIR)*, pages 142–150, 1994.
- [71] Tefko Saracevic. Relevance: A review of the framework for thinking on the notion in information science. *Journal of the American Society for Information Science*, 26:321–343, 1975.
- [72] L. Schamber, M. Eisenberg, and M. Nilan. A re-examination of relevance: Towards a dynamic situational definition. *Information Processing and Management*, 26:755–776, 1990.
- [73] Hinrich Schuetze. Personal correspondance.
- [74] A. Spoorri. Infocrystal: a visual tool for information retrieval and management. In *Proc. Information and Knowledge Management*, 1993.
- [75] Padmini Srinivasan. A comparison of the two-poisson, idf and discrimination value models of document representation. *Information Processing and Management*, 26(2):269–278, 1990.

- [76] Louise T. Su. Evaluation measures for interactive information retrieval. *Information Processing and Management*, 28(4):503–516, 1992.
- [77] J.A. Swets. Information retrieval systems. *Science*, 141:245–250, 1963.
- [78] R. Tong, A. Winkler, and P. Gage. Classification trees for document routing. In *Proceedings of the 1st Text Retrieval Conference (TREC-1)*, pages 209–228, 1993.
- [79] C.J. van Rijsbergen. *Information Retrieval*. Butterworths, 2nd edition, 1979.
- [80] Ellen M. Voorhees. Using wordnet to disambiguate word senses for text retrieval. In *Proc. 16th Int'l Conference on R&D in IR (SIGIR)*, pages 171–180, 1993.
- [81] Ellen M. Voorhees. Query expansion using lexical-semantic relations. In *Proc. 17th Int'l Conference on R&D in IR (SIGIR)*, pages 61–69, 1994.
- [82] Erik Wiener. Personal correspondance.
- [83] S.K.M. Wong and Y.Y. Yao. An information-theoretic measure of term specificity. *Journal of the American Society for Information Science*, 43(1):54–61, 1992.
- [84] David Yarowsky. Word sense disambiguation using statistical models of roget's categories trained on large corpora. In *Proceedings of COLING Conference*, pages 454–460, 1992.