The Relevance of Feedback for Text Retrieval

Vishwa Vinay

Department of Computer Science

University College London, London WC1E 7JE, U.K.
v.vinay@cs.ucl.ac.uk

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

of the

University of London.

Department of Computer Science

University College London

March 3, 2007

Declaration

I, Vishwa Vinay, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Thesis Committee

Primary Supervisor: Prof. Ingemar J. Cox, University College London
Secondary Supervisor: Dr. Mark Herbster, University College London

Internal Examiner: Prof. Mark Levene, Birkbeck College
External Examiner: Dr. Mark Sanderson, University of Sheffield

Abstract

Relevance Feedback is a technique that helps an Information Retrieval system modify a query in response to relevance judgements provided by the user about individual results displayed after an initial retrieval. This thesis begins by proposing an evaluation framework for measuring the effectiveness of feedback algorithms. The simulation-based method involves a brute force exploration of the outcome of every possible user action. Starting from an initial state, each available alternative is represented as a traversal along one branch of a user decision tree. The use of the framework is illustrated in two situations - searching on devices with small displays and for web search. Three well known RF algorithms, Rocchio, Robertson/Sparck-Jones (RSJ) and Bayesian, are compared for these applications.

For small display devices, the algorithms are evaluated in conjunction with two strategies for presenting search results: the top-D ranked documents and a document ranking that attempts to maximise information gain from the user's choices. Experimental results indicate that for RSJ feedback which involves an explicit feature selection policy, the greedy top-D display is more appropriate. For the other two algorithms, the exploratory display that maximises information gain produces better results. A user study was conducted to evaluate the performance of the relevance feedback methods with real users and compare the results with the findings from the tree analysis. This comparison between the simulations and real user behaviour indicates that the Bayesian algorithm, coupled with the sampled display, is the most effective. For web-search, two possible representations for web-pages are considered - the textual content of the page and the anchor text of hyperlinks into this page. Results indicate that there is a significant variation in the upper-bound performance of the three RF algorithms and that the Bayesian algorithm approaches the best possible.

The relative performance of the three algorithms differed in the two sets of experiments. All other factors being constant, this difference in effectiveness was attributed to the fact that the datasets used in the two cases were different. Also, at a more general level, a relationship was observed between the performance of the original query and benefits of subsequent relevance feedback.

The remainder of the thesis looks at properties that characterise sets of documents with the particular aim of identifying measures that are predictive of future performance of statistical algorithms on these document sets. The central hypothesis is that a set of points (cor-

responding to documents) are *difficult* if they lack structure. Three properties are identified - the clustering tendency, sensitivity to perturbation and the local intrinsic dimensionality. The clustering tendency reflects the presence or absence of natural groupings within the data. Perturbation analysis looks at the sensitivity of the similarity metric to small changes in the input. The correlation present in sets of points is measured by the local intrinsic dimensionality therefore indicating the randomness present in them. These properties are shown to be useful for two tasks, namely, measuring the complexity of text datasets and for query performance prediction.

Acknowledgements

The person who deserves the biggest thank you is Prof. Ingemar Cox. Sir, I will always take pride in knowing that I'm your first PhD student. I wrap up my studies with the sincere hope that the material in this dissertation is not all I've learnt in my time as your student.

Mum and Dad, I hope you are proud of your son. Now that I have finished the PhD, do you think you can try and figure out exactly what I've been doing for the past three years?

Prof. Mark Levene and Dr. Mark Sanderson, your critical comments during the defense will ensure that I think a lot more deeply about my work from here on in. At the same time, your encouraging remarks have provided me with the motivation and confidence to continue in research.

I owe a great deal to Microsoft Research Labs Cambridge, in particular Dr. Ken Wood and Dr. Natasa Milic-Frayling, for sponsoring my PhD and for all your advice right through my studies. But I have to sincerely question your judgment, now that you have decided to hire me!

Two people who won't let me get away alive if I don't put their names in here - Munni and Anita. Munni, I've won on technicalities but what was it that I get for winning the bet? And Anita, you know how much this meant to me, thanks for cracking the whip everytime I began to slack.

Thanks to all the people who have been members of the Ipswich gang over the years - Anuroop Shahi, Mohammed Ahmed, Andreas Pappas, Darren O'Shea, Matthew Iles, Catherine Fung, Ade Bamidele, Qiao Li, Lin Lin and Chin Wang. The last three years have been really important to me and you guys have made it fun.

University life can get frustrating because of a lot of administrative hassles. The staff at Adastral Park have been of great help by insulating me from all such problems and let me concentrate on work. Neil and Rich have helped a great deal by very kindly providing me with machines that I could then get to hammer away for my experiments.

Other people at UCL that I want to thank include Dr. Mark Herbster, Dr. Massi Pontil and Prof. Philip Treleaven.

Doing a PhD has been a childhood ambition and therefore it's not just about the last 3-odd years. Most of the people who've helped me in one way or another over the years will not be reading this. But I do wish to send out my thanks to each of them.

Contents

List of Figures

List of Tables

List of Symbols

$\mathbf{C}$ The collection of documents

N Number of documents in the collection

T Number of unique terms in the collection

d_i A generic document in the collection

$\mathbf{d_i}$ Representation of document d_i

t_{ij} Term frequency of term j in d_i

n_j Number of documents in $\mathbf{C}$ that contain term j

l_i Length of d_i

$\mathbf{q}$ Representation of user's query

K Number of terms in the query

$\mathbf{S}$ Result Set

$\mathbf{D}$ Displayed Set

$\mathbf{J}$ Relevance judgements for a query

$\mathbf{R}$ User labelled set of relevant documents for the query

$\mathbf{P}$ User labelled set of non-relevant documents for the query

n_R Number of documents in $\mathbf{R}$

n_P Number of documents in $\mathbf{P}$

r_k Number of documents in $\mathbf{R}$ that contain term k

n_D Size of $\mathbf{D}$

L Average length of documents in the collection

Chapter 1

Introduction

Easy access to computing resources has led to exponential amounts of information being generated over the past decade and this trend shows no signs of slowing down. Much of this information is stored for possible future use, some locally and some for public access. The networking of systems holding such information, along with the establishment of the required standards, has led to the introduction of the World Wide Web (WWW). Text documents are the most commonly found information on the WWW, though there is also an increasing proliferation of other media like images, audio, video, etc.

Due to the increasing sizes of these data collections, anyone seeking access to the information needs the assistance of automated mechanisms to find what they are looking for. Information Retrieval (IR) is the area that deals with the storage, organisation and access of information. The focus of this thesis is the retrieval of textual information, exemplified by the many web search engines available today. There has also been an increased availability of applications providing searching capabilities for locally held information, e.g. intranet and desktop search engines.

Such search engines accept a query, representing what the user is looking for, and provide a ranked list of potential answers. The goal of the IR system is to maximise the number of relevant documents in the ranked list as well as making sure that they are high up in the ranked list. The *quality* of the results is closely tied with the accuracy of the query. An inefficient query is often due to a vague information need but could also be because of an ineffective representation with respect to the information collection.

Relevance Feedback is a technique that assists in the automatic refinement of the query towards the goal of obtaining more relevant documents.

In an IR system offering Relevance Feedback (RF), the user is expected to provide relevance assessments for documents in an initial retrieval. This information is used by the system to alter the query such that the revised version is more *similar* to the marked relevant documents than the non-relevant ones.

First implemented by Rocchio [Roc71] for the SMART Retrieval system, RF has acquired different forms over the years. Commercial search engines have in the past experi-

mented with a “More like this” option for the users. In the academic environment, the focus has been on (explicit, and more recently implicit) labelling being provided by the user to narrow down the scope of the initial search. For example, when a user enters “jaguar” as the query, does he or she want information about the car or the wild animal?

There is a vast body of existing literature describing the results of experiments designed to measure the effectivness of adding relevance feedback to a retrieval system. However, the vote of confidence towards RF varies depending on the metric used for evaluation and the dataset and the query set considered for the experiments. For example, Salton [Sal70] and Robertson & Sparck-Jones [RSJ76] show very significant improvements with feedback on small test collections. However, Smeaton and van Rijsbergen [SvR83] report no improvements over a wide range of term weighting schemes when using the NPL test collection. Salton and Buckley [SB90], in a review of basic feedback procedures noted that, “Collections that perform relatively poorly in an initial retrieval operation can be improved more significantly in a feedback search than collections that produce satisfactory output in the initial search”.

Even though originally proposed in the context of text retrieval, relevance feedback has been more successful in the field of content based image retrieval (CBIR) [ZH03]. This is probably due to the fact that the time it takes for users to judge an image is much shorter than for a text document. Due to the success it has received in CBIR, a number of algorithms have been proposed to work with multimedia content and it remains an active research topic. However, this thesis will concentrate on the use of relevance feedback in the context it was originally designed for, i.e., text retrieval.

The standard text test collections, apart from providing a set of documents, consist of a query set and their associated relevance judgements. It has been observed (e.g. [ACR04, Kwo05]) that the success of query expansion and pseudo relevance feedback is dependent on the initial performance of the query. Salton and McGill [SM97], in describing the SMART system note that, “it has been shown experimentally that the relevance feedback process can account for improvements in retrieval effectiveness of up to 50% in precision for high-recall (broad) searches, and of approximately 20% in precision for low-recall searches”. Being able to estimate the difficulty of a query has wider implications, beyond its relation to RF, and is currently an active research area. Recent work [CYTDP06] has shown that some queries are inherently difficult as dictated by their relation to the data collection.

Text collections contain inherent regularities which retrieval algorithms attempt to exploit. The degree to which such regularities are present in the data could provide an indication of the success of future applications that work on this data. Understanding and explaining the regularities in the data can point towards factors that need to be accounted for while designing solutions. This coupling between a dataset and the effectiveness of various algorithms is reflected in the wide spread of results obtained by different teams participating

in the TREC initiative. Even for a given system, the results vary across the test collections.

This thesis provides suggestions for three specific issues - evaluation of relevance feedback, describing the complexity of a set of documents and query performance prediction.

Structure of the thesis

Before describing the original work of the thesis, Chapter 2 provides a review of existing literature while also introducing terminology that will be used in the rest of the dissertation. After providing a quick overview of text retrieval, the relevance feedback process is described and three algorithms that will be referred to in the rest of the thesis are introduced.

Chapter 3 provides the description of a simulation-based evaluation framework that is designed to provide a well-rounded view of a feedback algorithm's performance. It works by enumerating the entire space of user actions, thereby accounting for every possible user action. This strategy has the obvious problem of being expensive computationally and its application is therefore restricted to domains where it is feasible. Two such applications are considered and an experimental comparison of three algorithms introduced in Chapter 2 is provided.

Chapter 4 introduces three properties that can be used to describe quantitative properties of a textual dataset. These are the clustering tendency, sensitivity to perturbation and the local intrinsic dimensionality. Four subsets of the TREC data collection are compared on the basis of these properties and a relative ordering in terms of complexity (as indicated by each measure) is then provided.

In Chapter 5, these same properties are used to analyse the result set of a query. It is then shown that these properties are indicative of the performance of this query. The ability of these measures to serve as query effectiveness estimators is then evaluated on a standard TREC dataset.

Finally, Chapter 6 sums up what has been learned in this endeavour and suggests some directions for building on this work.

Chapter 2

Background

An Information Retrieval (IR) system helps users find information within a database. It usually does so by providing the user with a set of documents, known as the result set, in response to a user query. This broad definition of IR can be further refined into a number of more specialised sub-fields, which will be discussed shortly. However, before doing so, it is helfpul to define some basic concepts and terminology.

Each individual element in the database of the IR system is given the generic label 'document'. The most abundantly available form of such information is text documents and will be the primary focus of this thesis. The text of an email or a single web-page is an example of a document. Alternatively, the documents could be images, audio, video or multimedia documents.

Text documents consist of sequences of words. The IR system represents each document using a set of features. Typically, these features are terms, where 'term' has been used to mean one of three things:

1. A word in the document
2. Stemmed words that group together variations of a word (e.g. "computer", "computers" and "computing" to the same root which is "compute")
3. Key words that are identified by automatic/manual procedures

The database of documents is called a 'collection' and is the data source that is going to be searched during the IR process. A collection could for example be a personal collection of emails, a research group's publications or a very large collection of web-pages.

As mentioned earlier, the task of an IR system is to retrieve a set of documents in response to a user's information need. There is sometimes a distinction made between this need and its expression (i.e., what the system receives) which is known as a 'query'. In text retrieval systems, the query is expressed as a sequence of terms. The system uses a scoring function that associates a numerical value with each document in the collection indicating its *relevance* with respect to the current query.

Relevance is a central idea in IR. Mizzaro provides a survey of the different definitions of relevance found in IR research [Miz97]. Relevance is by nature subjective, therefore

making it difficult to automatically decide what is relevant and what is not.

Retrieving relevant answers in response to a query is the goal of "ad hoc retrieval", a subfield of IR. This thesis focuses on this task because apart from being the most often used form of IR, ad hoc retrieval provides a starting point for other tasks. All IR tasks are likely to contain an ad hoc retrieval component in the form of having to match a given document with a query (which could be another document). Alternative IR settings are discussed below.

- In the case of ad hoc retrieval, the data collection is comparatively static, while the information needs of the users are continually changing. A related situation is where the information need is constant and the collection is dynamic. The task of picking from a continuous stream those documents that match a user's profile is known as filtering. With every incoming item, the system has to make a choice as to whether or not it is to be delivered to the user. The user provides feedback to the system, helping it to refine its understanding of the user's need. The challenge is in designing algorithms that deliver maximum relevant information (and minimum non-relevant information) even with limited training data.

- Even though the task is referred to as 'information' retrieval, it is mostly just 'document' retrieval. The response to a query only indicates the particular documents which may contain the information relevant to the query. Question answering is an area which is a step towards providing the user with the actual information relating to his/her request.

- In all the above situations, the document and query languages were the same. Cross-language information retrieval deals with the problem of finding documents related to a query, regardless of which language the documents were originally written in. This allows multilingual searchers to issue a single query to a multilingual collection.

- Over the last few years, one form of Information Retrieval has received more exposure than all others - that of searching the World Wide Web. The size and dynamic nature of this collection and the fact that it includes a wide range of media types makes searching it a challenging task. Due to the unrestricted manner in which web pages are created, there is likely to be an inconsistency in the style of the actual content (text) in the pages - this is the reason why retrieval techniques for web search are tailored to take advantage of any available side information obtained from the structure of the web itself.

- If the aim is the organisation of digital information, classification and clustering are two important methods used to this end. While semantic labels can be attributed to the documents by the use of trained classifiers, clustering helps bring out the inherent structure in the dataset.

2.1 Components of an ad hoc retrieval system

Central to the operation of a retrieval system is being able to compare the query with each document in the collection. This is achieved by way of a scoring function, the input into which is a representation of the documents and the query. The particulars of the scoring function and the representation of the documents depends on the retrieval model being used (discussed in Section 2.2). Conversion of the query, from its input form to its representation, is performed online immediately after being entered by the user. For the documents in the database, the conversion is typically an offline process that is performed once (or at regular intervals for changing collections).

Consider a collection $\mathbf{C}$ containing N documents and T unique terms. Each document in this collection is denoted by d_i and computing the representation for each d_i is a one-way mapping $d_i \Rightarrow \mathbf{d_i}$. The d_i in its original form is a sequence of words. The representation $\mathbf{d_i}$ can be seen as a sequence of T weights which correspond to the degree to which each term characterises the document.

If each document in the collection is seen as a sequence of weights, the collection itself can be represented as a term-document matrix where the entry in row j column i is an indication of the importance of term j to document i. Populating the individual entries in the matrix is dependant on the particular retrieval model being used.

Once the pre-processing step is complete, the system is ready to accept user requests. A user submits a query to the system which, after an appropriate conversion to its representation, is compared against every document in the data collection. This produces a ranking that indicates the potential relevance to the query with documents achieving higher scores placed closer to the top. Using a threshold, either in terms of minimum score with respect to the query or a fixed number of desired results, the top few ranking documents are identified as the result set $\mathbf{S}$.

The result set is a list of candidate documents that are likely to address the user's query. A subset of the retrieved results are next returned to the user in the form of a displayed set $\mathbf{D}$. The size of $\mathbf{D}$ is dictated by constraints such as physical size of the display, user preference, etc. Conventionally, $\mathbf{D}$ is constructed by picking the top ranked documents of $\mathbf{S}$.

The term 'search' is used to describe one query-ranking-display cycle. By examining the documents in $\mathbf{D}$, the user might find what he/she is looking for. The user's query being successfully addressed indicates the conclusion of the current 'search session'. However, due to various factors, the displayed set is likely to contain a mixture of relevant and non-relevant documents.

Unsatisfactory results may lead to the user entering an alternative modified query. A level of interactivity is sometimes included as part of the system design to help in this query reformulation process. This technique is known as Relevance Feedback and will be the primary focus of this thesis and a separate section is devoted to the description of its various

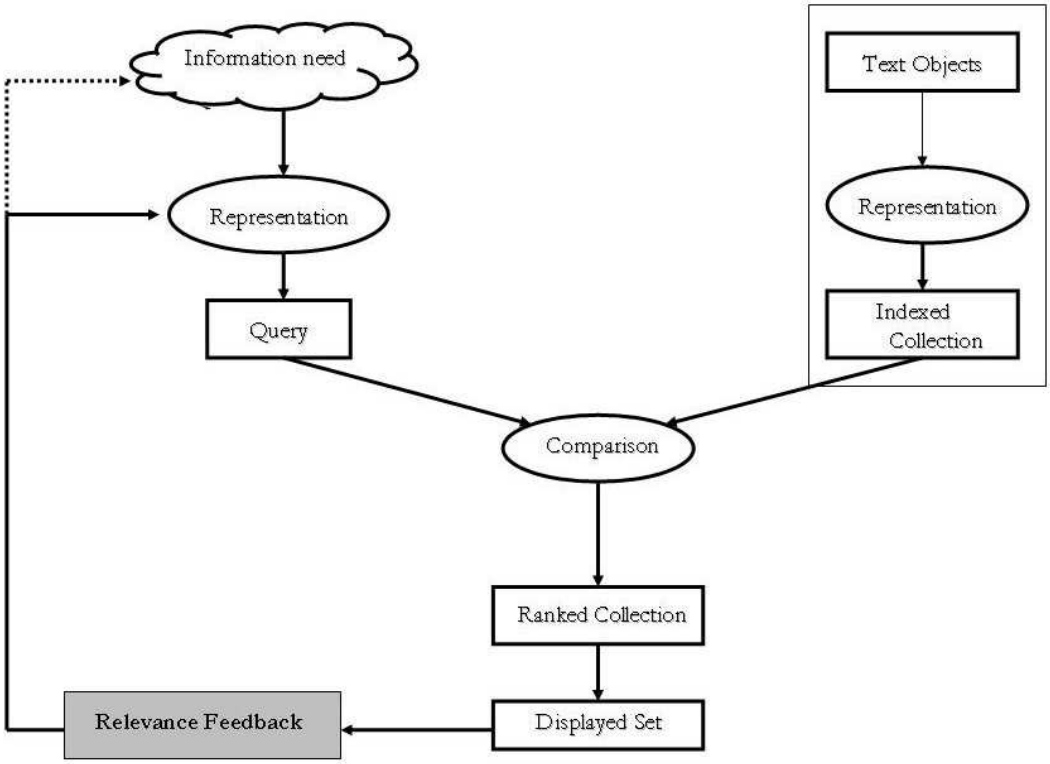

Figure 2.1: Information Retrieval

elements. It is hoped that the new query will produce a retrieval set that contains more relevant documents (and fewer non-relevant). This iterative process may continue for multiple cycles, the session ending with either a successful search or the user giving up.

Figure 2.1 provides the main components of an IR system for ad hoc retrieval.

2.2 Representation and Models

As mentioned in the previous section, comparing a document with a query requires both to be represented in a compatible form. A scoring function is used to assign to each document in the collection a value reflecting its relevance with respect to the current query. A few standard alternatives for choosing the representation and scoring function are described below.

It is common practice to use terms as the features to describe documents. To do so, each document needs to be individually parsed. Tokens, which are strings separated by whitespace or punctuation, are extracted by the use of a lexical analyser. This step, known as tokenisation, also converts strings to lower-case, extracts any meta-data that might be attached to the document and helps identify other elements like numbers, email addresses, etc.

After a document has been reduced to a series of tokens, stemming can be applied. This is the simplest example of morphological analysis and involves stripping the suffixes from

words in order to reduce variants of a word to the same root. Doing so reduces the number of unique tokens seen in the collection and has been shown to improve performance [Hul96]. The most common stemming algorithm is the Porter stemmer [Por80]. This stemmer uses a long list of rules, hand-crafted for the English language, which successively removes commonly occuring suffixes. A commonly used criticism of the Porter stemmer is that the output terms are not necessarily valid English words. Alternate stemmers include the rule-based Lovins stemmer and the dictionary-based Krovetz stemmer.

The output of the stemming algorithm are the terms which are used as the units for representation. Richer representations can be achieved by the use of natural language processing. Linguistically motivated features include phrases (contiguous words in the document), part-of-speech tags, etc. These features have had limited success and therefore are ignored for the rest of this thesis.

The next step is one of feature selection - picking a subset of terms that effectively represents the documents. A dictionary is constructed containing all the unique terms seen across the collection. The elements of this dictionary are arranged in decreasing order of number of occurances in the collection. Zipf's law in its original form states that in a collection of natural language documents, the frequency of a word is inversely proportional to its rank in the frequency table [Zip49]. Terms that are very frequent across the collection are unlikely to have any significance as they do not have any discriminatory value in identifying a document. Such terms, known as stop-words, are usually functional words (prepositions, conjunctions, etc.) in the language and are removed from the dictionary. At the same time, terms that are extremely rare across the collection are also disposed off as noise (e.g. typos). The features (terms) required to represent the objects (documents) have now been identified.

The terms can be considered to be features or attributes of the documents and there are a number of ways to assign values to them. Along with the scoring function used to compare a query to a document (or in fact one document to another), the weighting scheme is defined by the retrieval model being used. Some standard models are discussed below.

Boolean Model

Every document $\mathbf{d_i}$ in a collection with T unique terms and N documents can be seen as a vector $\mathbf{d_i}$ = (d_{i1}, d_{i2}, d_{i3},.., d_{iT}). If the elements of $\mathbf{d_i}$ are Boolean, i.e., 0 or 1, indicating the presence or absence of the word in this document, it is called the 'set of words' representation.

A user query is expressed in the form of a few select terms connected by Boolean operators (e.g. logical AND, logical OR). Precise queries can be difficult to formulate and therefore most systems use a default connector between terms (typically AND). Since we know which terms are present in each document, the scoring function for this model involves evaluating a series of set operations, finally producing a result set, every element of which satisfies the given Boolean query.

One drawback of this model is that since it uses set-based functions, the output will not be a ranking. This becomes much more of an issue when large sets of documents are returned, with no mechanism to differentiate one from the other in terms of potential relevance.

Vector Space Model

If a term occurs multiple times in a document, it indicates an increased significance of this term to this document. A binary representation will be unable to reflect this. Making the weight for term j in document i a function of the number of occurances of the term in d_i (i.e., the "term frequency") is the simplest form of a weighted representation. This is known as the 'bag of words' representation because positional information ("where does this word occur in this document?") is still discarded.

How important a given term is to a document is not only dependant on how frequently it occurs in the document, but also on how frequently the term occurs across the collection of documents. The inverse document frequency (idf) of a term is the reciprocal of the fraction of documents in the collection that contain this term. Combining the idf with the term frequency (tf) leads to the tf-idf model.

Terms in longer documents are likely to have larger term frequencies. In order to prevent this bias towards verbosity, the term frequency is normalised by the length of the document. The 'length' can be measured in a variety of ways, e.g. number of terms in the document, number of characters, maximum value of term frequency for this document, etc.

As in the Boolean Model, each document is represented as $\mathbf{d_i}$ = (d_{i1}, d_{i2}, d_{i3},.., d_{iT}). The weight d_{ij} of term j in document i is most commonly given by

$$d_{ij} = \left(\frac{t_{ij}}{l_i}\right) * log\left(\frac{N}{n_j}\right) \tag{2.1}$$

t_{ij} is the number of times term j occurs in document d_i and l_i is the length of the document, together comprising the 'tf' component. N is the number of documents in the collection and n_j is the number of documents in the collection in which word j is present, making up the 'idf' weighting.

Alternative weighting schemes exist, but they work on the same general principle - that of assigning a weight d_{ij} which indicates the contribution made by term j towards document d_i. Salton and Buckley investigated a range of weighting schemes and the results are shown in [SB88].

By using one of the weighting schemes, each document becomes a point in T-dimensional space. The query can also be thought of as a point in this space. The scoring function used for retrieval now reduces to associating a measure of proximity from the query to each document. For text retrieval, the commonly used measure is the cosine dot product. Normalising the vectors with their vector lengths leads to unit vectors such that the inner dot product between two such vectors gives us the cosine of the angle between them. It should

be noted that the length is different from the l_i used in Equation 2.1. The vector length of a document $\mathbf{d_i}$ is given by $\sqrt{\sum_{j=1}^{T} d_{ij}^2}$. The cosine similarity between two documents $\mathbf{d_i}$ and $\mathbf{d_j}$ is given by

$$\text{cosine}(\mathbf{d_i}, \mathbf{d_j}) = \frac{\sum_{k=1}^{T}(d_{ik} \cdot d_{jk})}{\sqrt{\sum_{t=1}^{T} d_{it}^2} \cdot \sqrt{\sum_{t=1}^{T} d_{jt}^2}} \tag{2.2}$$

The larger the value of the cosine product, the more similar the two vectors are. The measure can be used to compare either two documents or one document and the query.

The Vector Space Model(VSM) thus represents each document as a vector in a term space of very large dimensionality. The axes of this space are assumed to be orthogonal, thus representing an independence assumption for the terms. This is obviously a simplifying assumption as correlations exist between terms. Early papers (e.g. [WR84] and [WWY92]) provide discussions about the VSM. Due to its simplicity and performance that is comparable to most other models, the VSM remains a convenient framework in which to perform IR research. The VSM does not have parameters that need to be tuned for each collection thus making it more favourable.

Probabilistic Model

The root of the Probabilistic Model of IR is the notion of *relevance*, which is assumed to be a binary variable. Given a document $\mathbf{d_i}$ and a query $\mathbf{q}$, the quantity $P(R = 1 \mid \mathbf{d_i}, \mathbf{q})$ represents the probability of relevance conditioned on the query $\mathbf{q}$ and the given document $\mathbf{d_i}$. Using Bayes' rule, this quantity is modified as follows:

$$P(R = 1 \mid \mathbf{d_i}, \mathbf{q}) = \frac{P(\mathbf{d_i} \mid R = 1, \mathbf{q}) * P(\mathbf{q} \mid R = 1)}{P(\mathbf{d_i} \mid \mathbf{q})} \tag{2.3}$$

Similarly, the non-relevance given $\mathbf{d_i}$ and query $\mathbf{q}$ is given by

$$P(R = 0 \mid \mathbf{d_i}, \mathbf{q}) = \frac{P(\mathbf{d_i} \mid R = 0, \mathbf{q}) * P(\mathbf{q} \mid R = 0)}{P(\mathbf{d_i} \mid \mathbf{q})} \tag{2.4}$$

The Probability Ranking Principle [Rob97] suggests that the ranking of documents presented to the user should be based on $P(R = 1 \mid \mathbf{d_i}, \mathbf{q})$. In practice, the log-odds ratio $log(O(R \mid \mathbf{d_i}, \mathbf{q}))$ is calculated. Calculating the odds of relevance for the query $\mathbf{q}$ and document $\mathbf{d_i}$, $O(R \mid \mathbf{d_i}, \mathbf{q})$, also makes the practical task of implementation simpler because it negates the need to calculate the potentially complicated expression of document likelihood ($P(\mathbf{d_i} \mid \mathbf{q})$).

$$\begin{aligned} log\left(O(R \mid \mathbf{d_i}, \mathbf{q})\right) &= log\left(\frac{P(R = 1 \mid \mathbf{d_i}, \mathbf{q})}{P(R = 0 \mid \mathbf{d_i}, \mathbf{q})}\right) \\ &= log\left(\frac{P(\mathbf{d_i} \mid R = 1, \mathbf{q})}{P(\mathbf{d_i} \mid R = 0, \mathbf{q})}\right) \; + \; log\left(\frac{P(\mathbf{q}, R = 1)}{P(\mathbf{q}, R = 0)}\right) \\ &= log\left(\prod_{k=1}^{K} \frac{p_k^{b_{ik}} . (1 - p_k)^{1 - b_{ik}}}{\underline{p}_k^{\;b_{ik}} . (1 - \underline{p}_k)^{1 - b_{ik}}}\right) \; + \; log\left(\frac{P(\mathbf{q}, R = 1)}{P(\mathbf{q}, R = 0)}\right) \end{aligned} \tag{2.5}$$

where

b_{ik} is a Boolean variable which is 1 when term k is present in d_i and 0 otherwise

K is the number of terms in the query

p_k is the probability of term k occuring in a relevant document

$\underline{p}_k$ is the probability of term k occuring in a non-relevant document

Using the presence or absence of a term in the document to calculate the probability follows from the Binary Independence Model [RSJ76]. Mutual independence between terms is reflected by the fact that the probability can be calculated as the product of individual probabilities arising from each separate term.

$$\begin{aligned} log\left(O(R \mid \mathbf{d_i}, \mathbf{q})\right) &= \sum_{k=1}^{K} log\left(\frac{p_k^{b_{ik}}.(1-p_k)^{1-b_{ik}}}{\underline{p}_k^{\,b_{ik}}.(1-\underline{p}_k)^{1-b_{ik}}}\right) \;+\; log\left(\frac{P(\mathbf{q}, R=1)}{P(\mathbf{q}, R=0)}\right) \\ &= \sum_{k=1}^{K} b_{ik} * log\left(\frac{p}{\underline{p}_k}\right) \;+\; \sum_{k=1}^{K}(1-b_{ik}) * log\left(\frac{1-p_k}{1-\underline{p}_k}\right) \;+\; log\left(\frac{P(\mathbf{q}, R=1)}{P(\mathbf{q}, R=0)}\right) \\ &= \sum_{k=1}^{K} b_{ik} * log\left(\frac{p_k(1-\underline{p}_k)}{\underline{p}_k(1-p_k)}\right) \;+\; \sum_{k=1}^{K} log\left(\frac{1-p_k}{1-\underline{p}_k}\right) \;+\; log\left(\frac{P(\mathbf{q}, R=1)}{P(\mathbf{q}, R=0)}\right) \end{aligned} \tag{2.6}$$

The second and third terms in the above equation are independent of the document and are therefore going to be constant for the collection (given a particular query) and can thus be ignored when producing a ranking.

$$log\left(O(R \mid \mathbf{d_i}, \mathbf{q})\right) \propto \sum_{k=1}^{K} b_{ik} * log\left(\frac{p_k(1-\underline{p}_k)}{\underline{p}_k(1-p_k)}\right) \tag{2.7}$$

If the collection contains N documents of which n_k contain term k, and n_R have been labelled as being relevant to the current query of which r_k contain the term k, p_k and $\underline{p}_k$ can be approximated as follows:

$$p_k \approx \frac{r_k}{n_R} \quad \& \quad \underline{p}_k \approx \frac{n_k - r_k}{N - n_R} \tag{2.8}$$

Substituting these values, we get

$$log\left(O(R \mid \mathbf{d_i}, \mathbf{q})\right) \propto \sum_{k=1}^{K} b_{ik} * log\left(\frac{r_k}{n_k - r_k}.\frac{N - n_R - n_k + r_k}{n_R - r_k}\right) \tag{2.9}$$

The log term provides a weighting for each query term. The calculation of this weight ofcourse relies on the existance of a labelling of relevant documents (see Section 2.4.3 on the Robertson/Sparck-Jones feedback algorithm). It has been noted in [Rob04] that in the absence of this information, the expression reduces to $log\left(\frac{N-n_k}{n_k}\right)$, which is very similar to an idf weighting.

The Probabilistic Model of IR was championed by the Okapi system [RWHB$^+$95]. For ranking documents with respect to a given query, the system used the following weighting scheme:

$$w_k = log\left(\frac{(r_k + 0.5)}{(n_k - r_k + 0.5)}.\frac{(N - n_R - n_k + r_k + 0.5)}{(n_R - r_k + 0.5)}\right) \tag{2.10}$$

which is the same as the original weight, with the addition of a constant, 0.5, to ensure that none of the values are 0. For scoring, the BM25 function is used where the term-frequency

of a term k in document $\mathbf{d_i}$ is combined with the above weight using the combination

$$w_k * \left(\frac{t^c_{ik}}{t^c_{ik} + \underline{K}^c} \right) \text{where } \underline{K} = K1((1-b) + \frac{b * l_i}{L}) \tag{2.11}$$

t_{ik} is the number of times term k occurs in d_i, l_i is the length of d_i while L is the average length of documents in the collection. b and $K1$ are constants that control the non-linearity of the final score's dependence on the term frequency and document length and are parameters tuned for a given collection. c is another parameter of the system.

Language Modelling

The language modelling framework [Pon98] is a comparitively newer model that is gaining in popularity due its better performance in most tasks. Here, each document is assigned a language model which is a probability distribution over its terms. Rather than deal with the issue of relevance, this model produces the ranking of the documents based on the probability of generation of the query from a document's model.

The reasoning behind the Language Model is that the user has a particular document in mind and generates a query from this document. The quantity $P(\mathbf{d_i}, \mathbf{q})$ is then calculated as

$$P(\mathbf{d_i}, \mathbf{q}) = P(\mathbf{d_i}) \prod_{k=1}^{K} ((1-\lambda)P(k) + \lambda P(k\,|\mathbf{d_i}\,)) \tag{2.12}$$

which denotes the probability that the current document $\mathbf{d_i}$ is the one the user had in mind when generating $\mathbf{q}$. The number of words in the query is given by K. $P(k\,|\mathbf{d_i}\,)$ is the probability that the document will generate term k. In practice, this probability is replaced by its maximum likelihood estimate:

$$P(k\,|\mathbf{d_i}\,) = \frac{t_{ik}}{\sum_{k\prime}^{T} t_{ik\prime}} \tag{2.13}$$

A term k not present in the given document d_i will have $t_{ik} = 0$ leading to $P(k\,|\mathbf{d_i}\,) = 0$. This problem is precluded by introducing a smoothing parameter λ that allows terms to come from a general language model. One way of doing this, known as Laplace smoothing, involves having a pseudo-count for each term's presence in a document - $t_{ik} = \epsilon$ if it does not exist in the document and $t^{'}_{ik} = t_{ik} + \epsilon$ otherwise.

When the similarity between two documents $\mathbf{d_i}$ and $\mathbf{d_j}$ is to be calculated, the Kullback-Leibler (KL) divergence [CT91] is used. The KL function is the relative entropy between two probability distributions. It is 0 when the two distributions are the same and positive otherwise. For the case of two documents $\mathbf{d_i}$ and $\mathbf{d_j}$, the "distance" is given as follows:

$$D(\mathbf{d_j}, \mathbf{d_i}) = -\sum_{k=1}^{T} P(k\,|\mathbf{d_j}\,) log\,(P(k\,|\mathbf{d_i}\,))\; + \sum_{k=1}^{T} P(k\,|\mathbf{d_j}\,) log\,(P(k\,|\mathbf{d_j}\,)) \tag{2.14}$$

where $P(k\,|\mathbf{d_i}\,)$ is the probability of the language model of $\mathbf{d_i}$ generating term k.

2.3 Evaluation

As has been shown in the previous sections, there are various alternatives for each component of an IR system. In order to make a comparison between alternatives, objective measures of effectiveness need to be identified.

Criteria that evaluate the scalability of the system is one class of evaluation measures. The average query response time, for example, is a critical property for large scale retrieval engines that could potentially face very many simultaneous user requests. The size of the representation the system uses (along with any auxillary data structures the system needs to maintain) is also an important factor. The representation in any case should be much smaller than the data collection itself.

The Cranfield model [CMK66] defines a methodology for comparing different IR systems on the basis of their ability to retrieve information relevant to the query. The process begins with the construction of a standard common data collection against which retrieval is performed. A set of queries is constructed which acts as a substitute for real user queries. Assessors are recruited to provide relevance judgements that map each query to a set of relevant documents in the collection. Competing algorithms/systems issue the provided queries to the standard dataset and use their specific methodology to produce a result set for each query. The larger the number of relevant documents (as determined by the assessors) in the result set of each query, the more effective the retrieval.

Given the result set $\mathbf{S}$ for a query $\mathbf{q}$ with relevance judgements $\mathbf{J}$, the set-based Precision and Recall measures are the most common metrics used to measure effectiveness. Recall measures the ability of a retrieval system to display all the relevant documents and precision reflects the ability of the system to display only the relevant documents.

$$\text{Precision} \;=\; \frac{\text{Number of relevant documents in } \mathbf{S}}{\text{Total number of documents in } \mathbf{S}}$$

$$\text{Recall} \;=\; \frac{\text{Number of relevant documents in } \mathbf{S}}{\text{Total number of relevant documents in } \mathbf{J}}$$

Each query can thus be associated with a precision and recall value on this collection. In situations where a single metric is preferred, the F-measure is sometimes used.

$$\text{F} \;=\; \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

which represents the harmonic mean of the precision and the recall.

The precision-recall values can be calculated for each individual query, but in order to establish the performance of a retrieval system, it needs to be evaluated over a (large enough) set of queries. Performance of systems is sometimes reported in the form of a precision-recall graph. As can be expected, there is a trade-off between the precision and recall for a query - increasing one leads to a drop in the other. Therefore, the precision-recall graph is typically monotonically decreasing - as recall increases, precision decreses.

To be able to generate the precision-recall graph representing *average effectiveness* over a set of given queries, a method for aggregating individual query effectiveness needs to be defined. Since the number of judged relevant documents varies for each query, averaging across queries is done by interpolating precision values to a set of standard recall levels (0 to 1 at intervals of 0.1). The interpolation rule is to associate each standard recall level c with the maximum precision obtained for this query for any actual recall level greater than or equal to c.

Consider a result set **S** that contains n_R relevant documents, each at rank a_i, $1 \leq i \leq n_R$. For the set of documents starting from rank 1 upto (and including) the relevant document at a_i, calculate the precision ($\mathrm{Precision}(a_i)$) and recall ($\mathrm{Recall}(a_i)$) at each a_i. The interpolated precision at a standard recall level c is given by

$$\mathrm{Interpolated\_Precision}(c) \;=\; \mathrm{Max}(\mathrm{Precision}(a_i)) \text{ such that } \mathrm{Recall}(a_i) \geq c$$

The precision-recall curve is then the plot of $\mathrm{Interpolated\_Precision}(c)$ Versus c for $0 \leq c \leq 1$.

A measure commonly used as a summary for the quality of retrieval over a set of queries is the Mean Average Precision (MAP). The Average (non-iterpolated) Precision for each query is calculated by taking the mean of the precision scores after each relevant document is retrieved.

$$\text{Average Precision} \;=\; \tfrac{1}{n_R} \textstyle\sum_{i=1}^{n_R} \mathrm{Precision}(a_i)$$

The MAP for a set of queries is the mean of the individual average precision values.

Another precision-oriented measure is the $P@10$ which calculates the precision after 10 documents have been retrieved. Since most practical retrieval scenarios require that the top-few results contain the relevant documents, this measure asks how many of the top 10 documents are relevant to the query. The particular choice of cut-off, 10, is quite arbitrary and the precision at other thresholds (5, 100, etc.) are also considered.

All the measures described above were quantitative estimates of retrieval effectiveness. However, the end aim of a retrieval system is to satisfy the user's information need and therefore it is important to consider the user during evaluation. This is typically done by way of user trials wherein qualitative, and often subjective, properties of the retrieval system are measured. By their nature, such trials tend to be time-consuming and expensive but at the same time are indispensable at some stage of the process of system development.

2.4 Relevance Feedback

Users decide what constitutes "good performance" for a retrieval system. Since relevance is subjective, it is difficult to design retrieval functions that work across different users in different contexts. But, what makes relevance especially difficult to assess is that a user's need can change for every search and sometimes during a search session.

In an ideal retrieval system, the response to a query would be a set of documents that are relevant to the query, and nothing else. In actuality, depending on the quality of the initial

query, many documents may be retrieved but few may be relevant. “Relevance Feedback” is the collective term given to a wide range of techniques that attempt to address these issues by keeping the user involved (either actively or passively) in an interactive session of information seeking.

Conceptually, most of the systems that involve user Relevance Feedback (RF) can be described by a three-phase iterative process as depicted in Figure 2.2. This iterative process may begin with an initial query to the ranking engine, as depicted, or by a display of some selection of documents generated by the system itself. The feedback process consists of the following components:

1. Picking the display - In a system that includes RF, the set of documents displayed to the user at any given time serves two purposes. Firstly, to make sure that the user finds what he/she is searching for, this selection needs to be chosen so as to maximise the likelihood of containing the desired information. At the same time, if the required information is not found, any feedback indicated through a relevance labelling of documents in the displayed set needs to be most informative to the system in terms of reducing future search effort. These possibly contrasting aims mean that sufficient thought needs to be spent on this aspect.

2. Feedback interface - Depending on the *type* of feedback desired of the user, the interface should be designed appropriately. Documents in the displayed set can be labelled with relative or absolute relevance judgements which in turn can either be Boolean or on a real scale.

3. Re-ranking based on the feedback - Depending on the particular feedback algorithm being used, the relevance information provided is used to update some internal state of the system which then produces a fresh ranking of the documents in the database. This ranked list is then used to pick the next display of documents to the user. And the process repeats.

Despite the similarity of RF with Active Learning [Lew95] and a growing body of work in the semi-supervised learning area of machine learning, the question of picking feedback examples has received surprisingly little attention in literature. Conventionally, the top few ranked documents in the result set are used as the display set.

Considering that users may not be forthcoming in terms of providing feedback, the main restriction on the design of the interface is that it should be as simple as possible, from the users’ point of view. This means that typically Boolean judgements about the relevance of the displayed items is elicited, even though it is recognised that relevance is not a binary quantity. Recently, there have also been efforts towards the design of systems which capture factors describing the behaviour of the user (e.g. time spent reading a document, scrolling to the end, etc.) and interpret them as being implicit feedback information.

Traditionally, it is the re-ranking that has received the most attention. For feedback

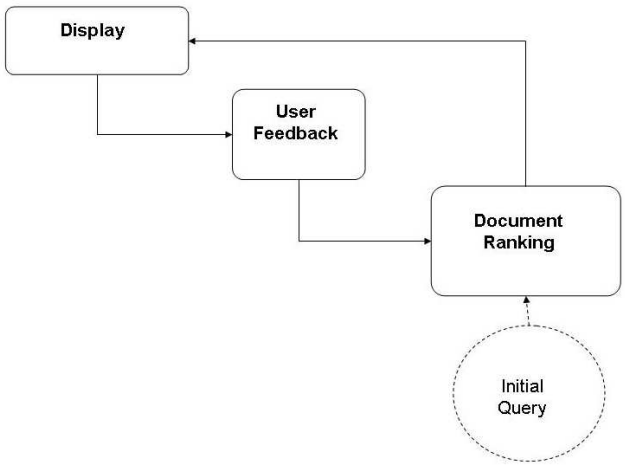

Figure 2.2: Relevance Feedback

algorithms dealing with text, any information provided by the user is used to alter the stated information need, i.e., the query.

The first component of the query reformulation process is query term re-weighting. Relevance information gathered from the user is used to adjust the relative importance of terms, based on their occurance in the documents labelled relevant and non-relevant.

The second component is that of query expansion. The initial query as provided by the user may be an inadequate representation of the user's information need, either in itself or in relation to the representations of the documents. Assuming that a longer query is more specific, the idea is to supplement the initial query with additional terms that help in retrieving other relevant documents.

A situation discounted in this thesis is where the user, dissatisfied with the retrieved results, manually issues a new query with additional (and sometimes deleted) terms. Some systems use knowledge-based mechanisms where outside sources (e.g. ontologies, thesauri) are used to append additional terms (e.g. synonyms) to the initial query in the first round of retrieval itself. "Relevance Feedback" only concerns the process where the set of documents returned by an earlier retrieval are examined to provide clues as to candidate supplemental query terms. The degree of user involvement in the identification of terms to be added defines a spectrum of RF techniques.

On one end of the scale is Interactive Query Expansion (IQE). Statistics from the displayed set and the data collection as a whole are used to pick candidate terms/phrases, which are then provided as part of the user interface. Some subset of these are chosen by the user to be the new additional terms. This expanded query then induces a new ranking on the data collection from which the next display is picked.

An alternative to IQE is Automatic Query Expansion (AQE) where the chosen terms, rather than being displayed to the user, are automatically added to the query. In the case

of the vector space model, the updated query contains all the terms present in the relevant documents (see the Rocchio algorithm below). In the probabilistic models, the terms seen so far are ranked on the basis of some criterion, and the top few are chosen (see the Robertson/Sparck-Jones algorithm).

Diametrically opposite to IQE on the scale of interactivity is the practice of pseudo-relevance feedback [CH79]. The assumption here is that documents ranked high in an initial retrieval are relevant by default. The system uses the top ranking documents as positive examples without the user explicitly labelling them. The re-ranked list produced by the RF algorithm is used to pick the first display set presented to the user. The success of this method obviously relies on the presence of relevant documents high in the initial ranking and is sometimes appropriately referred to as “blind relevance feedback”.

2.4.1 Justifying Relevance Feedback

After 30 years of research in Information Retrieval, even the best system has limited recall. A few of the relevant documents, but not all, are retrieved in response to the intial query. Users are still unsure of the best way to formulate an initial query to a search engine. It is fair to assume that the majority of users start with a simple query and then react to what the system does.

Researchers have experimented with various algorithms to determine the most successful formula for calculating relevance feedback. While the details of their experiments vary depending upon the methods tested, one result remains consistent: an improvement over baseline searching is seen if relevance feedback is included.

Defining the degree of interactivity required to produce better results is an active research topic. This question has been extensively tested with respect to query expansion in particular. As mentioned in the earlier section, in automatic query expansion (AQE) the system uses a pre-defined criterion to add terms to the query before producing the revised results. The users are unaware of the query expansion. In interactive query expansion (IQE), the users have some control over which terms are added.

Three types of interfaces are possible for an interactive retrieval system using relevance feedback:

1. Opaque - The feedback process is a black box
2. Transparent - The terms added to the query are displayed as part of the interface
3. Penetrable - The list of candidate terms are present to the user who can choose which ones they want added to the query

Koenemann and Belkin [KB96] conducted an experiment in which novice users interacted with the system through each of these interfaces. Despite being reluctant to fully use the control provided by the penetrable interface, the users of this system did the best overall.

Similarly, Anick [Ani03] notes that while interactive relevance feedback is seen to be

effective, the practical aspect of implementation is difficult because users are reluctant to make the document relevance judgements. Given users' reluctance to interact, retrieval systems are typically less interactive and allow the computer to do most of the work.

Many of these problems may be alleviated by the use of implicit feedback indicators [WRJ02]. Instead of requiring the user to provide explicit judgements about a document's relevance, substitute features like document reading time, mouse movement, scrolling, etc. are used as *interest indicators* which provide evidence of relevance. Such features are likely to be less accurate than explicit feedback, but provide an alternative mechanism that is sensitive to user behaviour.

Without making a distinction between explicit and implicit feedback, the question is, can the effect of adding feedback to a retrieval system be estimated conclusively? The aim of this thesis is not to present a new algorithm, but it is to identify the situations (as described by the properties of the retrieval process) when a feedback iteration is likely to be beneficial. Most studies of feedback have reported positive results, and this sort of exploration of the factors affecting the success of feedback could lead to the design of algorithms that use this information more efficiently.

A comprehensive review of relevance feedback can be found in [Har92] and [SB90]. A survey of the use of relevance feedback in information access systems is provided in [RL03]. In the next section, specifics of three RF algorithms are provided. The Rocchio and Robertson/Spack-Jones algorithms were chosen as representatives of the vector space and probabilistic models respectively. Both these algorithms have been in use for text retrieval for many years. In contrast, the Bayesian algorithm described here was originally designed for content-based image retrieval.

2.4.2 The Rocchio Algorithm

The Rocchio relevance feedback scheme [Roc71] is used in conjunction with the term-frequency inverse-document-frequency (tf-idf) representation where documents and queries are represented as vectors of term weights and similarity is measured by the cosine distance between these vectors. A document is a vector $\mathbf{d_i}$ = (d_{i1}, d_{i2}, d_{i3},.., d_{iT}) where T is the number of terms across the collection. A query $\mathbf{q}$ = (q_1, q_2, q_3,.., q_T) is defined similarly. A weight is associated with each term in the query. The weights are similar to the term-document weights and indicate the importance of each term to the query. The evidence provided by the user (in the form of relevance judgements) is used to specialise the relative weighting of the query terms to this particular session. The documents and queries are normalised for length by setting

$$\mathbf{d_i}' = \frac{\mathbf{d_i}}{\|\mathbf{d_i}\|} \text{ and } \mathbf{q}' = \frac{\mathbf{q}}{\|\mathbf{q}\|} \text{ where } \|\mathbf{x}\| = \sqrt{\sum_{\mathrm{j}=1}^{\mathrm{T}} \mathrm{x_j^2}}$$

The similarity score between document $\mathbf{d_i}$ and query $\mathbf{q}$ is then given by the cosine similarity as given in Equation 2.2.

$$\text{score}_{\text{rocchio}}(\mathbf{d_i}, \mathbf{q}) = \frac{\sum_{k=1}^{T}(d_{ik} \cdot q_k)}{\sqrt{\sum_{t=1}^{T} d_{it}^2} \cdot \sqrt{\sum_{t=1}^{T} q_t^2}} \tag{2.15}$$

The Rocchio algorithm takes a set $\mathbf{R}$ of relevant documents and a set $\mathbf{P}$ of non-relevant documents (as selected in the user feedback phase) and updates the query weights according to the following equation:

$$w'_k = \alpha w_k + \beta \frac{\sum_{r \in \mathbf{R}} d_{rk}}{n_R} + \gamma \frac{\sum_{p \in \mathbf{P}} d_{pk}}{n_P} \tag{2.16}$$

where n_R and n_P are the number of relevant and non-relevant documents respectively.

The parameters α, β, and γ control the relative effect of the original weights, the relevant documents, and the non-relevant documents.

Rocchio is provided here as an example of an algorithm used in conjunction with the vector space model. Two variations of the query update formula in (2.16) are described by Ide in [Ide71]:

$$\text{Ide} : w'_k = \alpha w_k + \beta \sum_{r \in \mathbf{R}} d_{rk} + \gamma \sum_{p \in \mathbf{P}} d_{pk} \tag{2.17}$$

This is the same as the Rocchio formula without the normalisations.

$$\text{Ide Dec} - \text{Hi} : w'_k = \alpha w_k + \beta \sum_{r \in \mathbf{R}} d_{rk} + \gamma \underline{P}_k \tag{2.18}$$

$\underline{P}_k$ is the weight of term k in the highest ranked document labelled as being non-relevant, $\underline{P}$. Experiments by Ide indicate that the Dec-Hi method provides best improvements in retrieval effectiveness.

2.4.3 The Robertson/Sparck-Jones Algorithm

The Robertson/Sparck-Jones algorithm [RSJ76] is used in conjunction with the probabilistic model of information retrieval. The terms in a collection are all assigned relevance weights which are updated for a particular query whenever relevant documents are identified. Initially the relevance weights are given idf-based values, the weight for term k is given by $w_k = \log(N/n_k)$, where N is the total number of documents in the corpus and n_k is the number of documents containing term k. A document $\mathbf{d_i}$ is assigned a score against query $\mathbf{q}$ based on the relevance weights of the query terms occurring in each document.

$$\text{score}_{\text{rsj}}(\mathbf{d_i}, \mathbf{q}) = \sum_{k \in \mathbf{q}} \frac{(K1 + 1) * t_{ik}}{K1((1 - b) + \frac{b * l_i}{L}) + t_{ik}} * w_k \tag{2.19}$$

where

t_{ik} is the number of occurrences of term k in document d_i

$K1$ and b are parameters of the algorithm

l_i is the length of document d_i

L is the average length of all documents in the corpus

This is the same as described in Equation 2.11 with $c = 1$.

Documents are then ranked in descending score order. If certain documents are flagged as relevant, the relevance weights are updated as given in Equation 2.10 as follows:

$$w_k = \log\left(\left(\frac{r_k + 0.5}{n_k - r_k + 0.5}\right)\left(\frac{N - n_k - n_R + r_k + 0.5}{n_R - r_k + 0.5}\right)\right) \tag{2.20}$$

where

r_k is the number of relevant documents containing term k

N, n_R and n_k are defined as before

In addition to updating the relevance weights, the relevant documents are used to select new (or additional) query terms according to some criterion that relfects their potential utility. This feature selection component of the feedback algorithm results in query expansion. Many term selection methods have been proposed with the probabilistic retrieval model [SJWR00]. The offer weight, sometimes referred to as the Robertson Selection Value (RSV), is one such example. This offer weight o_k for a term k is given by o_k = r_k * w_k where r_k and w_k are as defined in (2.20). The quantity o_k is a measure of the potential utility of term k as a candidate query expansion term. All the terms are ranked in decreasing order of their offer weights and the top terms are used as part of the subsequent query. How many such terms are to be chosen per iteration is another parameter of the system.

There have been other attempts at defining feature selection methods, but the general consensus is that they all provide equivalent performance in terms of retrieval effectiveness across a range of different queries. In [Eft93], Efthimiadis looked at six different ranking algorithms for the case of IQE and attempted to measure the agreement of real users with the list of suggested terms provided by the algorithm. The results indicate that even though there are sometimes differences in the terms selected by different algorithms, the resulting improvement in retrieval performance and user satisfaction with the provided options are all comparable. The encouraging fact was that given a ranked list of options, the terms scoring high according to the selection criterion were almost always the ones the users chose.

2.4.4 The Bayesian Algorithm

The Bayesian relevance feedback algorithm was first proposed for a Content-Based Image Retrieval System called PicHunter [CMM+00]. It is assumed that the user is searching for a particular data item d_U, known as the user's target document, that is present in the collection. The recursive probabilistic formulation associates with every document, at each iteration, z, the probability, P_z of document d_i, being the target document, $\mathbf{d_U}$. This probability is conditioned on all current and past user actions and the history of displayed documents, which collectively is denoted by H_z. The concept of a current query, $\mathbf{q}$, is not explicitly present in this formulation. Thus, at each iteration, the document rankings are given by

$$\begin{aligned} \mathrm{score}_{\mathrm{bayesian}}(\mathbf{d_i}) &= P_z(\mathbf{d_i} = \mathbf{d_U} \,|H_z\,) \\ &= P_{z-1}(\mathbf{d_i} = \mathbf{d_U} \,|H_{z-1}\,) * G(\mathbf{d_i}, \mathbf{R}) \end{aligned} \tag{2.21}$$

where P_{z-1} is the document's probability in the previous iteration
$\mathbf{R}$ is the set of documents marked relevant in this iteration

$$G(\mathbf{d_i}, \mathbf{R}) = \prod_{r \in \mathbf{R}} \frac{\exp\left(\frac{sim(\mathbf{d_i}, \mathbf{d_r})}{\sigma}\right)}{\sum_{((j \in \mathbf{D}) and (j \notin \mathbf{R}))} \exp\left(\frac{sim(\mathbf{d_i}, \mathbf{d_j})}{\sigma}\right) + \exp\left(\frac{sim(\mathbf{d_i}, \mathbf{d_r})}{\sigma}\right)} \tag{2.22}$$

where $\mathbf{D}$ is the displayed set and $\mathbf{R}$ is the set of documents labelled as being relevant in this iteration.

The term sim($\mathbf{x}$, $\mathbf{y}$) computes the similarity of document $\mathbf{x}$ with document $\mathbf{y}$, which for textual documents can be taken as the cosine dot product of tf-idf vectors normalised for length. The tuning noise parameter σ is set according to the specific dataset. The algorithm works by increasing the probability $P_z(\mathbf{d_i} = \mathbf{d_U})$ of document $\mathbf{d_i}$ being the target document if it is closer (according to the cosine similarity) to documents that have been marked as being relevant in previous iterations.

While the Rocchio algorithm was designed explicitly for the vector space model and the RSJ algorithm for probabilistic retrieval, the Bayesian algorithm is purely a relevance feedback algorithm that is independent of the retrieval method used. The choice of the cosine dot product as the similarity measure however makes it closer to the vector space model. While other similarity measures (e.g. the BM25 ranking function) can be used, the cosine product has many desirable properties including being bounded between 0 and 1.

The difference between the Bayesian algorithm and the Rocchio and RSJ algorithms is the absence of a query. User feedback, in the form of a relevant/non-relevant labelling, is utilised to update the probabilities, but there is no "query expansion". The update of the probability over the documents in the collection is dependant on the similarity metric, which implicitly uses all the features. The initial query is used to provide a starting probability distribution across the documents in the collection. Alternatively, the system might start with a prior distribution biased towards more *popular* elements. It should be remembered that no document should be given a 0 initial probability (even if it does not contain a single query term) because it would then never have a non-zero probability, regardless of any relevance information that is subsequently provided.

2.5 Evaluation in Interactive IR

This section discusses only the quantitative measures of the effectiveness of an interactive retrieval. Other measures, which typically can only be estimated by exhaustive user trials, would need to consider the users' experience of the interface. The amount of interactivity exposed to the user and the ease with which the various features can be exploited to benefit the user are subjective qualitative quantities that will not be dealt with in this thesis.

Designing experiments that reflect the true potential (and highlight the inadequacies) of the interactive algorithms has received much attention in IR literature. Early papers dealing with this subject (e.g. [HW67]) stressed the need for caution when using precision and recall

directly for interactive IR evaluation. At the heart of the problem was the understanding that documents labelled as being relevant (i.e., the ones used in the RF cycle) will necessarily be retrieved again after feedback, and normally will occupy higher ranks. Since the user has already evaluated the documents involved, this improvement is not particularly important and it is the position of $unseen$ relevant documents that is important.

As an alternative, a method known as *freezing* was suggested [CCR71]. A set of documents $\mathbf{D}$ is displayed to the user, who then provides relevance information for elements of this set. After re-ranking, a new display set $\bar{\mathbf{D}}$ is produced, which does not contain the elements of $\mathbf{D}$. Precision and recall statistics are calculated on the concatenation of $\mathbf{D}$ and $\bar{\mathbf{D}}$. Williamson in [Wil78] provides the reasoning and the effects of freezing.

Testing along the lines of the training/test set divisions have also been proposed. The given collection is split into two groups, the control and test group. Documents for RF are taken from an initial retrieval on the test group and the modified query is issued against the control group on which the precision-recall statistics are calculated. However, without an even representation of relevant documents in both the sub-collections, the measures calculated run the risk of being unrepresentative of the actual performance.

A technique known as "Incremental Feedback" [Aal92] has been proposed which has a twin advantage of not only exposing a very simple interface to the user, but being simpler to evaluate. The interface displays a single document at every iteration, which the user then labels as being relevant or not. The information is then used by a standard feedback algorithm to produce a fresh ranking. Previously judged relevant documents are maintained as part of the interface. The *search effort* is represented by the number of judgements made by the user and is used as a criterion for evaluating the benefits of feedback.

Evaluation based on the Cranfield model does not transfer very easily into situations where interaction, of varying complexities, are part of the system. Simulation-based methods for evaluation have been proposed as an alternative.

To test the effectiveness of the probabilistic re-weighting formula, which is based on feedback from the user, Sparck-Jones used a few relevant documents to estimate the weights and measured its effect on the remaining documents [Jon79]. This can be seen as an early simulation-centred experiment to evaluate pseudo-relevance feedback. To measure the benefits that interactive query expansion could offer, Harman simulated a *good selection* of expansion terms by making a reasonable approximation that the candidate terms will all be present in the unseen relevant documents. More recently, [MvR97] and [Rut03] employ very similar methods for comparing automatic and interactive query expansion and reach the conclusion that automatic feedback could potentially offer large benefits but real users find it difficult to take maximum advantage of the technique. White (e.g. [WJvRR04]) also makes use of simulated work tasks [Bor03] in his work on the use of implicit feedback measures.

In the experiments discussed in Chapter 3 of this thesis, a framework for measuring the effectiveness of relevance feedback algorithms is presented. The method is also simulation-based and employs a brute-force strategy of examining the outcome of every possible user action, thereby providing a more rounded view of each feedback algorithm. This methodology is then used to compare the three feedback algorithms described earlier in this chapter.

Chapter 3

Relevance Feedback Evaluation based on Exhaustive Enumeration

Interactive information retrieval can be difficult to evaluate. User trials are typically the solution but they can be time consuming and expensive. In this chapter, a methodology for the evaluation of feedback algorithms is discussed and is used for the comparison of three standard algorithms.

For evaluation, *performance* needs to be clearly quantified and this varies depending on the type of search. What constitutes a successful search depends on what is being searched for. In the context of retrieval, at least three classes of search may be identified [CMM+00]:

1. Target document search - the user's information need is satisfied by a particular document. For example, a researcher may be looking for a specific paper on a research topic.
2. Category search - the user seeks one or more items from a general category or a topic. This task places more emphasis on the content evaluation and often requires subjective relevance judgements.
3. Open ended browsing - the user has some vague idea of what to look for but is open to exploration and may repeatedly change the topic during search.

Of these three scenarios, the target document search is most amenable to evaluation as there are several clear measures of effectiveness. Though this task sounds rather restrictive, it encomposses a wide spectrum of search scenarios. On the one hand there is 'known item search' which represents a subset of target-search situations where the user is familiar with the target document. For example, a user wants to re-visit a website, but having forgotten the URL, types keywords about the website into a search engine. On the other hand, the user can be unfamiliar with the target but a single document could exist in the collection that satisfies his/her information need. So long as the user can recognise that his or her information need is satisfied when a specific document is displayed, the scenario can be modelled as target document search.

Of particular interest is whether the relative performance of relevance feedback versus plain searching can be predicted without recourse to user studies. For example, the search effort without relevance feedback can be defined as the number of documents the user has to scroll through in an initial search. This can be compared with the the total number of documents presented and examined for the benefit of a feedback algorithm before a target is found.

Within a given search session and for a given RF algorithm, the number of judgements required to find a target is dependant on which documents are displayed to the user and which of these the user labels with relevance judgements. The effectiveness of the feedback algorithm is therefore closely tied in with the user's choices. This leads to the idea of the enumeration of the entire search space. The advantage of this methodology is that two important quantities can be measured, an upper bound on the effectiveness of the applied relevance feedback as well as an average case performance indication.

Analysis of the complete search space is an experimental paradigm that can lead to interesting insights into the behaviour of relevance feedback algorithms. The actions of a *real user* will already be part of this analysis. It also allows a large number of experiments to be performed and statistics that might be used to predict the actual user performance can be collected.

There are of course computational considerations regarding the use of this method, such an analysis is tractable only if the enumeration of all possible user actions (and their corresponding effects on the retrieval process) is possible. Typically, this is dependant on the feedback model - what sort of interaction is expected with the user? If this interface is very rich, in terms of providing the user with many alternatives, the enumeration will be very large. Since higher interactivity is not necessarily preferred by users, design of interfaces with limited interactivity offers the twin advantage of being easy to use (from the end-user perspective) and being easier to evaluate (from the system designer's perspective).

In this chapter, two important application domains where exhaustive enumeration is feasible are identified:

1. Searching on small display devices: Due to size constraints, the number of items displayed on the screen is small. This means that apart from situations when the initial user query leads to the desired document(s) being returned in the first retrieval, the user is likely to have to go through multiple screenfuls of results before finding what he/she is looking for. Rather than simply pressing the 'Next' button, if the user is asked to indicate one of the displayed documents as being relevant, the number of alternatives available to the user is equal to the number of items displayed on the screen. This makes searching on devices with small displays a realistic scenario for the use of multiple iteration feedback, which is typically not used in conventional retrieval scenarios.

2. Web search: Due to the size of standard desktop monitors most search engines use a default display size of 10, i.e., the results to any query are shown in groups of ten. Any combination of these can be marked as being relevant (leading to 2^{10} alternatives). However, only one iteration of feedback is considered since users typically do not go past the first couple of pages of results. Also, the web offers sources of evidence for the content of a page apart from the text it contains and their utility for RF can be investigated.

The enumeration methodology is used to compare three standard feedback algorithms for the two scenarios mentioned above:

1. The Rocchio Algorithm
2. The Robertson/Sparck-Jones(RSJ) Algorithm
3. The Bayesian Algorithm

all of which have been described in Chapter 2. The first two algorithms are chosen as being representatives of the vector-space and probabilistic models of information retrieval respectively. The Bayesian algorithm was designed for use with images and had not been previously applied to text.

The three algorithms also serve as alternatives of various complexities and methodologies. RSJ employs an explicit feature selection policy (for query expansion) while Rocchio uses all terms from the relevant documents. The Bayesian algorithm does not have an explicit formulation of a 'query'. This also means that when employing the inverse index lookup (which is typical of text retrieval), Rocchio and RSJ will be comparitively cheaper computationally (the query contains few terms and only documents that contain query terms are affected). For the Bayesian algorithm, the probability associated with every element in the collection needs to be updated at every iteration making it more expensive than the other two.

3.1 Interactive Retrieval on Small Display Devices

The continuing evolution of portable computing and communications devices, such as cell phones and Personal Digital Assistants (PDAs), means that more and more people are accessing information and services on the Internet with devices that have small displays. This small display size presents challenges. First, a need for extensive scrolling makes viewing of standard pages very difficult. Second, the input modes on PDAs or mobile phones are far less efficient than keyboard typing and make even a simple task of sending a text query rather time consuming. Finally, devices like mobile phones still lack computing resources and speed to perform sophisticated processing on the client side.

Of particular concern are the implications that small display devices will have on searching online information resources. Generally, it has been observed that users engage in a variety of information seeking tasks, from "finding" a specific, well defined piece of

information, to "gathering information" as a more open-ended, research-oriented activity [RMFSB03]. Use of Internet enabled mobile phones is still in its infancy and no general patterns of use have been established. Anticipating that mobile users will search for specific, well-defined information, this section concentrates on trying to understand how relevance feedback, display strategies, and other interactive capabilities can support users engaged in searching for a target document or piece of information.

A considerable body of research has been dedicated to the issues related to user interaction [JM97, JMMN$^+$99], browsing [BGMPW00, BGMP00], searching [SMS02, RMFSB03], and reading [CMZ03] on mobile devices. Most directly relevant to the study described here is Toogle [Ruv03], a front-end desktop application that post-processes Google results based on the user's actions. Toogle collects evidence that the presented documents are relevant or non-relevant documents from the user's clicks on documents in one or more screens of search results. It uses this information and machine learning techniques to re-rank the remaining documents. In contrast, the experiments describe here focus on searching using mobile devices when the user feedback is constrained to the selection of a single relevant document from a small number of documents presented at each iteration.

The experiments take advantage of the small display size and limited user actions to study the full space of the user's interactions and all possible outcomes determined by the relevance feedback and display strategies. As part of the simulation the 'ideal' user's actions can be identified and provides an upper bound on the performance of relevance feedback systems for small displays.

There are several research efforts that share some aspects of this approach. The interactive nature of the task makes it similar to the Ostensive Retrieval Model [CvR96]. Ian Campbell proposes that the information need of a user progressively changes based on the what has already been presented to him/her. Evidence, like the relevance labelling of documents in earlier iterations, is given a temporal weighting that associates an uncertainity with each feedback depending on how many interactions ago this information was provided. In the set of experiments described here, standard relevance feedback algorithms are used thereby assuming that all evidence has an equal contribution to the current state of the query.

Very recently, White *et al* [WJvRR04] measured the performance of implicit feedback models by conducting a simulation-based evaluation. The authors were interested in using factors like document read time, scrolling behaviour, etc. as feedback indicators rather than explicit judgements of document relevance that are used here.

The use of a single document as feedback, which the system then uses to automatically infer a new ranking over the data collection, has been previously studied by Aalbersberg [Aal92]. The evaluation framework proposed here extends Aalbersberg's use of a single document in the display to one where the user is provided with multiple items in every iteration, but expect only binary feedback regarding the relevance of one chosen document

from the displayed set.

3.1.1 Display Strategies

At each iteration, it is necessary to provide a display set $\mathbf{D}$ to the user. This set is to be chosen from the result set $\mathbf{S}$ and the most obvious strategy is to display the set $\mathbf{D}$ of documents with the highest rank in $\mathbf{S}$. After successive query refinements (i.e., multiple iterations of feedback), this Top-D display is likely to result in a set of documents very similar to one another. If these documents are similar to the target document or even include it, then this may well be optimum. However, if they are not similar to the target document, the user feedback is unlikely to help redirect the search away from the displayed documents and towards the target.

In the study of feedback over multiple iterations described in [Har92], it was shown that the number of relevant documents retrieved in later iterations reduced continually. This was attributed to the fact that some queries had all their relevant documents retrieved in earlier iterations. While this is partly true, it is also likely that the greedy nature of the display update leads to a form of overlearning or overfitting. This problem has been previously discussed in the context of content-based image retrieval [CMM$^+$00] and observed in the current experiments - a further description is provided in the section titled ‘Convergence’.

As has been discussed in [Lew95], relevance feedback can be seen as a form of active learning. The user can only label those documents that are displayed to him/her and this display set is chosen by the system. The dual aim of reducing the number of judgements required from the user and still being able to retrieve the relevant content at the earliest could potentially be achieved by choosing the right documents to be displayed/labelled. In active learning scenarios, the examples typically shown to the user are those that the system is most unsure about. Since the primary aim in these situations is the learning of the classifier, the ambiguous data items chosen for labelling can belong to either the positive (’relevant’) or negative (’non-relevant’) class. However, in information retrieval, there is a penalty for displaying non-relevant documents. Therefore, there should be a bias towards elements of the positive class.

Ideally, the display set should contain those documents for which a user’s response would be most informative to the system to minimise the number of search iterations. This was proposed by Cox *et al* [CMM$^+$00] and formulated as a problem of finding a set $\mathbf{D}$ of documents that maximises the immediate information gain from the user’s response in each iteration. Determining such a document selection is computationally expensive.

However, it can be approximated by sampling the elements of $\mathbf{D}$ from the underlying similarity score distribution using computationally efficient methods. For example, the sampling method may simulate a roulette wheel with the size of each item’s field proportional to the relevance score of a document with respect to the query.

Also known as ‘fitness proportional selection’, picking elements for the display set using

the probabilistic method can be implemented as follows. Firstly, all the documents in the collection are ranked according to their scores with respect to the query. Each document's score is then normalised by the sum of scores of all documents across the collection. After normalisation, individual document scores will lie between 0 and 1 and the cumulative score of all items in the collection will be equal to one. Next, a uniform random number generator is used to sample a number between 0 and 1. Starting with the top ranked document, the procedure moves down the ranking while maintaining the cumulative score of documents seen so far. The document whose score pushes the cumulative score above the random number generated is selected to be part of the display set. If n_D items are to be displayed, n_D different random numbers are generated, each leading to an element being added to $\mathbf{D}$.

Within such sampled displays both documents with high and low ranks have a non-zero probability of being included. The more peaked (certain) the ranking, the less likely it is to display low probability documents. Thus the display exhibits more variability and enables the user to direct the search away from a local maximum. It is expected that the sampled display strategy will be useful in situations where the initial query is imprecise, i.e., when the target document is ranked very low in the search result list. This method for picking the display set is referred to here as the "Sampled display".

Using devices with small displays for search raises issues similar to those encountered in adaptive information filtering where the importance of the interplay between exploitation and exploration has been recognised. It is to be expected that there are various sampling strategies that optimise the balance between exploitation and exploration. Finding the optimal selection of documents from which to gather feedback from the user (while still maximising the probability that it contains what the user is looking for) is an open problem. The probabilistic sampled display update provides a semi-principled and computationally efficient method for achieving this end.

3.1.2 Evaluation Methodology

In order to examine the effect of relevance feedback and alternative display strategies, an experimental procedure that included the complete space of possible user interactions with the system was devised. For a given query or information need, the user decision tree representing all possible document selections in each feedback iteration was created. This was feasible because of the small number of documents, n_D, that are displayed in each iteration. Thus, all user feedback strategies can be examined, including those of an 'ideal user' whose selection of documents minimises the number of documents that must be viewed before retrieving the target document.

In each iteration the tree expands by a factor of n_D (see Figure 3.1), i.e., the number of documents in an individual display. For practical purposes, the number of iterations were limited to five; the initial display of n_D documents followed by five iterations of relevance feedback. This results in a tree of depth five where each node corresponds to one display of

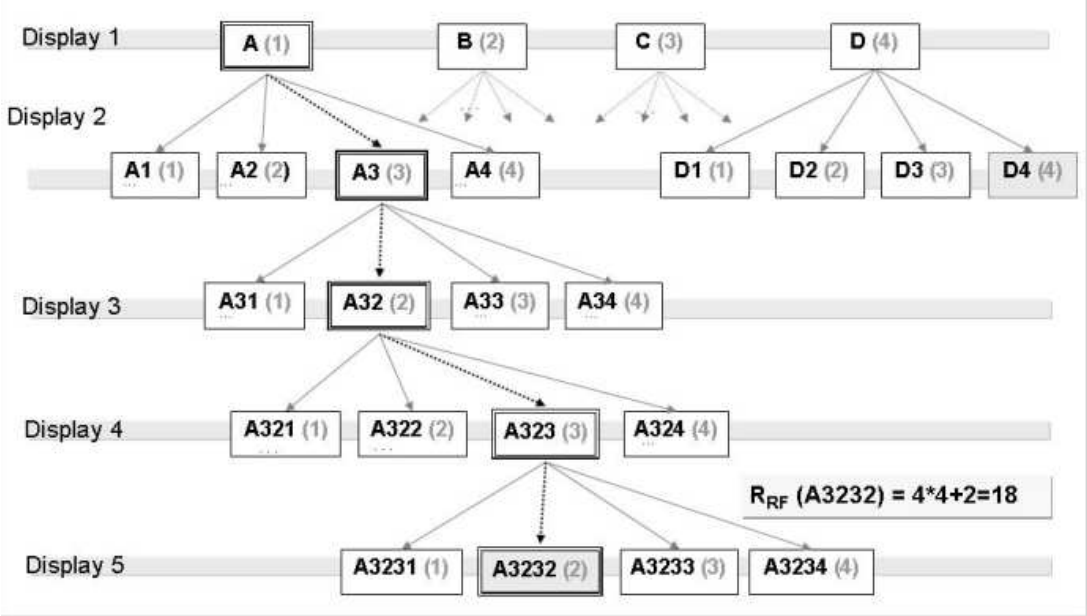

Figure 3.1: Decision tree for iterative relevance feedback, showing nodes in which the target document is reached, the rank of a document within each display, and the calculation of RF-rank for the target document labelled A3232

n_D items. In such a tree, after z iterations of feedback, the maximum number of nodes in the tree can be given by $(1 + \sum_{i=1}^{z} n_D^i)$, where the 1 corresponds to the initial display. For $n_D = 4$, the maximum number of nodes in the tree is $1 + 4 + 4^2 + 4^3 + 4^4 + 4^5 = 1365$, where a node represents a display of n_D documents. The tree may be smaller if the target is located earlier since the branches of the tree were not expanded once the target had been displayed. The choice of display size $n_D = 4$ was motivated by the size of a typical mobile device display. However, the same method could be used to investigate the effect of a range of display sizes.

The minimum rank for a given target document corresponds to the case when the user always provides the system with the optimal document for relevance feedback. It is important to note that 'optimal' may not always mean the document most similar to the target. Another factor to be examined is the number of occurrences of the target document in the decision tree. This provides an estimate of the likelihood that a non-ideal user will locate the target document. For example, if the target document appears in only one path of the tree, then any deviation of the real user from the relevance feedback of the "ideal user" would result in a failed search. Conversely, if the target document appears in many paths, then the deviations from the "ideal" are still likely to yield successful searches, albeit that these searches require further effort.

Since the trees are generated automatically, it is possible to create trees for a large number of searches, thereby facilitating a statistical analysis of the algorithms.

Construction of the user decision trees

Figure 3.1 provides an example of the user decision tree. At each iteration the tree expands by a factor of n_D=4. While the general behaviour of relevance feedback algorithms is of interest, from the application point of view it is most important to understand the impact of the first few iterations of relevance feedback. It is unlikely that the users would engage in a large number of feedback iterations. Therefore the tree expansion is limited to depth five, considering the root of the tree as depth zero.

The initial display of four documents is labelled A-B-C-D and is followed by five iterations of relevance feedback. At each iteration, selection of a document from the display leads to a new branch in the tree. Some of these branches would contain the target document. Since the evaluation was performed in the target search framework, the trees were not expanded below nodes that contained the target document.

Each document in the tree was annotated with its rank p within the display of $n_D = 4$ documents, with p having the value $p = 1, 2, 3,$ or 4. Displays from relevance feedback iterations were concatenated by appending to the list the most recent display. The resulting list shows documents in the order in which the user would view them. For each document in the tree, the corresponding ranked list is identified and the relevance feedback rank R_{RF} given by $R_{RF} = z * n_D + p$ was calculated. R_{RF} essentially corresponds to the number of documents that the user has viewed before locating the document at the p^{th} position after z iterations of feedback ($z = 0, 1, 2, 3, 4$ or 5). In the evaluations, R_{RF} was compared with the rank of the document in the initial search. This baseline rank is referred to as the scroll rank, R_{Scroll}, since this is the number of documents that the user would have to examine by scrolling down the original list of search results in order to reach the target document.

Initialisation

The experiments began by randomly selecting a target document from the database. An initial query is then automatically generated by randomly selecting K terms from the target document. $K = 4$ was used for the experiments. These K terms are used in two ways: as a search query to obtain the baseline search results and as input to the relevance feedback procedure which will further refine the query based on the user's responses.

Randomly sampling for query terms does not simulate query generation by users. Rather, it provided a method for analysing performance against queries of varying quality - a good query is indicated by the target occupying a position high up in the initial ranking, i.e., before relevance feedback is applied. Similarly, a bad query is indicated by the target occupying a position low down in the initial ranking. The value $K = 4$ was also chosen for the same reason. With higher values of K, since the sampled query terms were definitely present in the target, the target document was almost always ranked high in the initial retrieval. Smaller values for K led to a majority of the sampled queries being 'bad'. This value of K is therefore dependant on the collection being used and should be chosen

appropriately.

The user is initially shown a display of n_D documents that are chosen based on which display strategy is being used. The user's response is used by the relevance feedback algorithm to modify the query. The documents in the collection are then scored against the new query and a new display of n_D documents is presented to the user, based on the search ranking and display strategy. Previously viewed documents are not included in the subsequent search iterations.

Even after expanding the entire tree, the target may not be found. The quality of the initial query is a factor that determines whether or not the tree contains the target. Those trees where the target appears in atleast one path are referred to as "successful searches".

3.1.3 Dataset

The retrieval experiments need to be performed against a set of textual documents, the Reuters-21578 collection (Appendix A) was used for this purpose. The collection is a set of Reuters newswire items. Each document comes with SGML markups which amongst other things provide each document with a date of creation, a tag identifying a general topic categorisation, a "Title" section identifying the headline and a "Body" of text. The text from the "Body" and "Title" fields were extracted. After the removal of standard stopwords, the documents were converted to their respective representations. The documents with empty "Body" fields were removed, leading to a dataset of $19,043$ documents.

3.1.4 Experiments and Results

100 distinct target documents were randomly selected from the $19,043$ documents of the Reuters collection. The same set of 100 documents were used as targets across the 3 algorithms. The initial query was generated from a sample of terms occurring in the target document and the scroll rank of each target document was recorded.

For each target document a search tree based on iterative feedback was generated with two types of displays: (1) the Top-D display always showing the top 4 ranked documents from the search iteration and (2) the Sampled display that probabilistically selects the documents based on the current ranking of documents in the database. Tables 3.1-3.4 summarise the statistics of the tree displays and successful searches.

The scroll rank of a target document is the position of the document in the initial ranked list of search results, i.e., the number of documents that the user would have to scroll through in order to reach the target (in the absence of feedback). The RF rank of an ideal user is the minimum path length from the root of the tree to a node with the target, whereas the mean length of all paths leading to the target represents the average performance of successful users. The first row in Table 3.1 is the probability that a search (using a given display scheme) will be successful, and row two is the probability that a user will find the target given that this is a successful search. Non-ideal users correspond to all the choices in successful trees that lead to the target document but the path-length to the root (i.e., the

	Rocchio Feedback Algorithm		RSJ Feedback Algorithm		Bayesian Feedback Algorithm	
	Top-D	**Sampled**	**Top-D**	**Sampled**	**Top-D**	**Sampled**
Percentage of trees with target	52	97	39	33	52	90
Percentage of paths containing the target	46.67	4.5	27.99	0.087	46.80	4.30
Average R_{Scroll} of targets found in trees	13.79 (7.09)	98.54 (21.63)	37.28 (10.84)	312.03 (726.37)	7.92 (1.59)	64.23 (11.33)
Average min R_{RF} of targets found in trees	6.5 (0.99)	11.25 (0.68)	7.20 (1.10)	17.76 (2.64)	6.13 (0.92)	10.61 (0.73)
Average R_{RF} for the 'average user'	20.53 (682.83)	20.2 (149.02)	20.22 (624.87)	18.26 (3.75)	21.27 (816.91)	19.94 (143.30)

Table 3.1: Search tree statistics for the three feedback algorithms and two display strategies. The results are averaged over 100 searches for random targets

Scroll Rank Range	**Number of Targets**	**Number of Targets Found**		**Avg. No. of Documents Viewed without RF**		**Avg. No. of Documents 'ideal user' with RF**		**No. of Docs viewed with RF averaged over successful users**	
		Top-D	**Sampled**	**Top-D**	**Sampled**	**Top-D**	**Sampled**	**Top-D**	**Sampled**
$1-20$	45	45(100%)	45(100%)	4.58	4.38	4.31	5.33	16.54	19.13
$21-40$	14	6(42.8%)	14(100%)	25.5	29.79	20.67	13.07	21.62	21.92
$41-60$	5	0(0%)	5(100%)	–	54.2	–	16.6	–	21.99
$61-80$	4	0(0%)	4(100%)	–	66.5	–	16.5	–	21.80
$81-100$	6	0(0%)	6(100%)	–	92.83	–	15.33	–	21.49
>100	26	1(3.84%)	23(89%)	367	341.3	20	18.56	20.78	22.14

Table 3.2: Performance of the Rocchio RF Algorithm based on the Initial Query

Scroll Rank Range	Number of Targets	Number of Targets Found		Avg. No. of Documents Viewed without RF		Avg. No. of Documents 'ideal user' with RF		No. of Docs viewed with RF averaged over successful users	
		Top-D	Sampled	Top-D	Sampled	Top-D	Sampled	Top-D	Sampled
$1-20$	27	27(100%)	7(25.9%)	5.67	4.72	4.26	17	19.21	18.67
$21-40$	6	2(33.3%)	2(33.3%)	34	31	7.5	17	12.46	17
$41-60$	5	3(60%)	3(60%)	47.33	41.67	6.33	17.33	7.4	17.3
$61-80$	8	1(12.5%)	3(37.5%)	74	68.33	17	21	18.15	21
$81-100$	2	1(50%)	2(100%)	81	88	24	17	24	17
>100	52	5(9.6%)	16(30.7%)	187.2	606	18.2	17.5	21.72	17.94

Table 3.3: Performance of the RSJ RF Algorithm based on the Initial Query

Scroll Rank Range	Number of Targets	Number of Targets Found		Avg. No. of Documents Viewed without RF		Avg. No. of Documents 'ideal user' with RF		No. of Docs viewed with RF averaged over successful users	
		Top-D	Sampled	Top-D	Sampled	Top-D	Sampled	Top-D	Sampled
$1-20$	45	45(100%)	45(100%)	4.38	4.38	4.31	5.02	16.54	18.75
$21-40$	14	6(42.8%)	14(100%)	25.17	29.78	17.67	13.07	22.21	21.35
$41-60$	5	0(0%)	5(100%)	–	54.2	–	13.4	–	21.52
$61-80$	4	1(25%)	4(100%)	64	66.5	17	18.5	18.05	21.98
$81-100$	6	0(0%)	6(100%)	–	92.83	–	18.33	–	22.18
>100	26	0(0%)	16(61.5%)	–	254.56	–	18.44	–	21.92

Table 3.4: Performance of the Bayesian RF Algorithm based on the Initial Query

RF rank) is higher than that for the ideal user. For the Top-D display strategy, about 50% of the trees contain the target (lower for RSJ). In the remaining cases, the target was not found within five rounds of relevance feedback. This percentage is clearly a function of the accuracy of the initial query, which can be judged by examining the scroll rank of the target document. This will be discussed further.

The ideal user represents the best possible performance achievable. Real users are unlikely to perform as well. However, the average number of paths in the tree that contain the target suggests that deviations from the ideal still have a reasonable chance of locating the target document. The average rank of target documents in the tree was obtained by calculating first the average rank for the target document within its particular tree and then averaged over the set of all the trees that contain target documents.

Top-D Display Scheme

For the Rocchio and Bayesian algorithms, for a scroll rank of less than 20 (Tables 3.2 and 3.4, rows corresponding to scroll rank range $1 - 20$), relevance feedback with Top-D display is successful 100% of the time. For higher values of the initial scroll ranks, i.e., poor queries, a fall off in the percentage of successful searchers was observed. However, the sampled display approach offers performance that is more constant. For the case of RSJ, with an explicit term expansion strategy, the Top-D display performs better.

Convergence

It was observed that sub-trees below a node at depth 4 were often identical. That is, the set of four documents displayed to the user at depth 5 was the same, irrespective of the choice of relevant document at the preceding level. Note that the relative order of displayed four documents may be affected by the relevance feedback, but the same documents appeared in all four sub-trees. It is important to note that the convergence was observed for all three algorithms, even though the sets to which they converged were different.

Since the phenomenon was not symptomatic of any one particular algorithm, it was suspected that this convergence was due to the greedy nature of the display updating strategy - that of picking the n_D most probable items (based on the score with respect to the current query). Since the aim of the RF algorithm is to extract similar documents from the collection, it results in a situation where successive displays offer no diversity. This could be seen as a direct consequence of the "cluster hypothesis" [vR79] which states that documents relevant to the same query are likely to be similar to each other. The small variation across the documents in the display is also due to the small number of documents, 4, in the display.

Sampled Display Scheme

For the alternative display, a higher percentage of the trees contained the target document with the Rocchio (an increase from 52% for Top-D to 97% for Sampled) and Bayesian schemes (52% to 90%, refer Table 3.1). More importantly, a performance degradation as the

quality of the initially query degrades was not observed. And for very poor initial queries, the alternative display strategy was superior. Since the RSJ algorithm itself considers exploring different regions of the search space by query expansion, use of the sampled display strategy led to an over-adventurous approach, resulting in a smaller number of successful searches and fewer paths leading to the target in a given tree. This illustrates the classical dilemma between exploration and exploitation.

Analysis of the trees containing the target revealed that the average scroll rank was much higher than the rank for an ideal user using relevance feedback and the alternative display, representing a very significant reduction in the number of documents examined. However, once more, it needs to be recognised that real users are unlikely to perform as well as the ideal user.

For the sampled display, the average number of paths in the tree that contain the target is low, which would suggest that deviations from the ideal may have a significant detrimental effect on performance. The number of real users finding the target in the user trial that was conducted (Section 3.1.6) when using the sampled display, though lower than when using the Top-D display, does not however reflect this expectation. This would strengthen the case for the usage of the sampled display update. Finally, it is noted that the convergence phenomenon observed with the Top-D display was not exhibited using the sampled display.

Discussion

Rather than using the experimental framework described in the earlier sections to pick the best relevance feedback algorithm, the simulations help draw attention to certain aspects of the behaviour of RF algorithms. For example, row one of Table 3.1 can be used as a statistic to measure the performance of a RF algorithm and display update combination. It would be desirable to be able to reach the target for every initial query. Therefore a high number for the percentage of trees with the target will indicate a capable algorithm. But it would not be particularly helpful if every tree contained the target but only one path in the entire tree led to it. This is because it would mean that anything apart from one specific sequence of actions will not lead to a successful search.

The percentage of paths containing the target (row 2 of Table 3.1) is therefore equally important. It is not clear however what value a *good* algorithm will have for this statistic. Consider a tree in which every path led to the target, an extreme example of the convergence phenomenon described above. Though the tree will have 100% of paths leading to the target, it would indicate a particularly unresponsive feedback algorithm that is not sensitive to the user choices. And as mentioned above, if only one path leads to the target, this is again not desirable because only the ideal user will be able to find the target. A value somewhere in the middle will indicate an algorithm where a non-ideal user has a good chance of finding the target at the same time making sure that random searches are not rewarded with success.

If the first two rows of Table 3.1 are interpreted as being probabilities, a multiplicative

combination of the two values can be seen as being an *a priori* estimate of the probability of a user finding the target within five iterations of feedback. The Rocchio and the Bayesian algorithms with the Top-D display update have this probability equal to 0.24 while the RSJ algorithm with the sampled display has the lowest value of 0.0002.

For the Rocchio or the Bayesian algorithm, a quarter of all searches led to the target. This would seem to indicate that both algorithms, when used with the greedy display update, are extremely successful. However, a look at Tables 3.2 and 3.4 reveals that these algorithms were only able to find the targets for the most trivial cases, i.e., when the target was initially ranked very high. If the experiments were repeated with an alternate query generation scheme, e.g. picking the least informative terms from the target to generate the query ("bad queries"), these algorithms would be extremely unsuccessful. In contrast, even though the *a priori* probability of finding the target using the Rocchio and Bayesian algorithms with the sampled display is much lower, only 0.04, it should be noted that this combination was able to find the target across a range of initial scroll ranks.

In fact, we can use the variance of the initial R_{Scroll} (the quantity in parenthesis in row 3 of Table 3.1) as an indicator of the range of the initial query quality (e.g. good, mediocre, poor) over which an algorithm is successful. For the Rocchio and the Bayesian algorithms, the variance for R_{Scroll} is very small, indicating that the targets which were found in the trees came from a very small range of initial query quality. A larger value of the variance of R_{Scroll} for the sampled display and these two algorithms indicates that this display update strategy was able to find the target almost independent of the quality of the initial query.

Another observation that can be made based on the variance information is the fact that the ideal user can potentially achieve a consistent level of improvement. This is represented by the very low values for the variance of minimum R_{RF}'s found in successful trees. This means that *for the ideal user*,all the algorithms can guarantee a very high degree of success with large confidence.

The quantity "average minimum R_{RF} of targets found in trees" is indicative of the upper bound on the performance benefit of using RF. In all the trees that contained the target, by how much was the ideal user able to reduce the search effort (measured in terms of number of documents examined) when using relevance feedback? To calculate this improvement, subtract the average minimum R_{RF} (row 4 of Table 3.1) from the average R_{Scroll}. This number shows that both the Rocchio and the Bayesian algorithm achieved improvements of over 80% when using the sampled display scheme. This is, of course, the best possible improvement, while the more realistic *average improvement* is much lower.

This discussion reveals that selecting the best combination of relevance feedback and display update algorithms is not straightforward. However, by exploring the entire user space, it is possible to identify useful characteristics of RF algorithms. Conversely, using a more restricted model to simulate the user that only explores certain regions of the trees

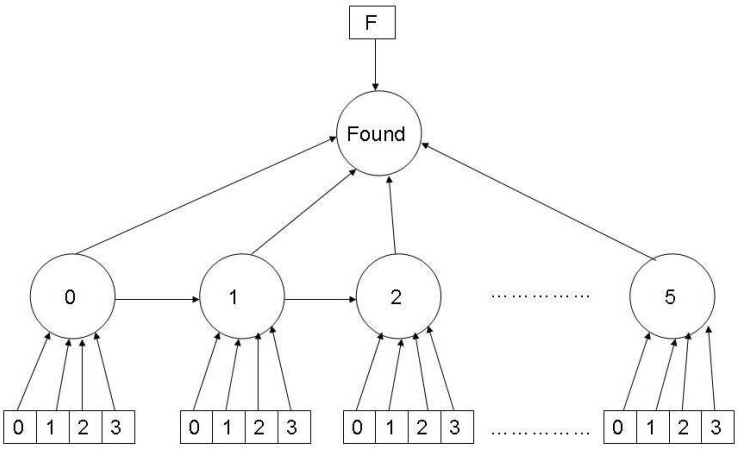

Figure 3.2: Tree paths represented as state changes

would not have highlighted these issues. Similarly, only relying on user trials gives an incomplete view on the performance of the algorithms, since each user only follows one path in each tree.

It should be noted that the specific set of results reported here are likely to be dependent on the choice of dataset and parameters used. However, the evaluation framework is sufficiently general to be used in other situations and on other datasets.

3.1.5 Constructing a Statistical Model of the "Successful Users"

The simulation-based framework outlined above provides a method for automatically investigating the effects of every possible user action. Some of these actions were successful (in terms of leading to the target) and most were not. The trees generated provide a data source that can be mined to produce a probabilistic model of the "successful users". This is similar to the construction of probabilistic automaton for navigation in hypertext described in [LL99].

Searching on a small display device with relevance feedback as described in the earlier sections involved the user examining the current screenful of (four) results and indicating one amongst them as being most relevant to his/her search. This information was then used to producing a fresh ranking over the items in the data collection from which a new display set was picked according to the chosen display strategy. The choices available to the user at any given iteration are limited and control what is going to be displayed in the next iteration. Such a system can be modelled as a Markov chain where moving across displays can be seen as state changes and probabilities can be associated with each available user choice for the current iteration.

Because of the availability of efficient algorithms for training and calculating probabilities of sequences of action, a Hidden Markov Model was used. The HMM had H hidden states and O output symbols. Here H is the number of displays as dealt with in the trees

plus an additional "Found" state. Moving from one display to another after an iteration of feedback is therefore a transition from one internal state to the next. This is therefore an 'absorbing HMM' that always ends in the "Found" state. However, unlike other applications of HMMs, there is nothing *hidden* since the iteration number is always available.

From each of the states corresponding to a display, $(O-1)$ of the allowed outputs can be generated - in this case, these $O-1$ are each of the possible user actions. The final O^{th} output is only allowed from the Found state. For the current experiments, $H=7$ (the initial display, five iterations of feedback and the "Found" state) and $O=5$ (choose one of four documents or being in the "Found" state). The model was built such that from a given display state, the only allowed transitions are into the next display state, or to the Found state. The diagrammatic representation is provided in Figure 3.2.

From the trees that were collected, the sequence of paths representing the choices that led to a successful search were extracted. Ignoring the searches where the target was found in the initial display, the remaining paths were used as training data for the HMM. The trained parameters of the HMM have the following interpretations:

1. The transition matrix is an estimate of finding a target in a given iteration (a transition to the Found state) against having to move onto the next iteration
2. The emission matrix indicates the optimal choice of 'relevant document' in a given display state.

For each of the 6 variations (3 RF algorithms * 2 display strategies), two sets of trained models were constructed:

1. Using all successful paths - representing the 'average user'
2. Using only the shortest path from each tree - representing the 'ideal user'

In the Transition Matrices both the rows and columns correspond to iteration numbers or display states, whereas in the Emission Matrix the rows correspond to the iteration and the columns represent the choice of relevant document in that iteration with the last column being the 'Found' state. It is the Emission Matrix in each case which is of interest. As an example here, the Emission matrices for the model of the 'average user' using the Bayesian feedback algorithm is provided (Table 3.5).

If in the Emission Matrices of the trained models, the first column dominated every row, this would strengthen the belief in the practice of choosing the highest ranked item in every iteration for feedback, i.e., pseudo-relevance feedback. On the other hand, a uniform distribution across the choice of relevant document (columns 1 to 4 all being 0.25) would indicate the absence of any significant pattern. However, some deviations from both these extremes was observed. For example, in almost all cases, with the Top-D strategy, there seems to be a preference for the lower ranked items (higher values in later columns, indicating the need for 'exploration'). But in the sampled display update scheme, there is a very small bias towards the higher ranked items.

Top-D Display				
0.16	0.31	0.28	0.25	0
0.24	0.27	0.21	0.28	0
0.17	0.26	0.28	0.29	0
0.25	0.28	0.23	0.24	0
0.29	0.29	0.23	0.19	0
0.26	0.23	0.23	0.28	0
0	0	0	0	1
Sampled Display				
0.26	0.24	0.25	0.25	0
0.26	0.26	0.24	0.25	0
0.26	0.24	0.25	0.25	0
0.28	0.25	0.24	0.23	0
0.33	0.28	0.20	0.19	0
0.56	0.26	0.13	0.05	0
0	0	0	0	1

Table 3.5: Emission Matrices for the trained model for the Average User using the Bayesian Algorithm. The columns correspond to the choice of relevant document and the rows are successive display states

It is not clear if they deviate enough from the uniform distribution to warrant being classified as interesting. However, it is another example of how the evaluation methodology can be used to gather other properties which can be used to design the system. A trained HMM is thus the statistical model of all "successful users" across the 100 trees that were built. A possible use of such a model would be for pseudo-relevance feedback: in a given state, the document(s) to be fed back implicitly as being relevant can be picked by the columns in the emission matrix with the highest values. The next section describes a user trial that was conducted to validate the user model.

3.1.6 User Trial

To test if the simulation-based framework corresponds in any way to the behavior of actual users, a small scale user trial of 12 subjects, all of whom were CS/EE PhD students, was conducted.

The user-interface consisted of a screen divided into two sections. The left half, running along the height of the screen, was used to display the 'target' continuously throughout the session. The users were given time to familiarise themselves with this target before proceeding. The right half of the screen was divided into four quadrants, each displaying one of four documents. At each iteration, the user was instructed to indicate the document most relevant to the target by clicking on it. A "Next" button was provided to move to

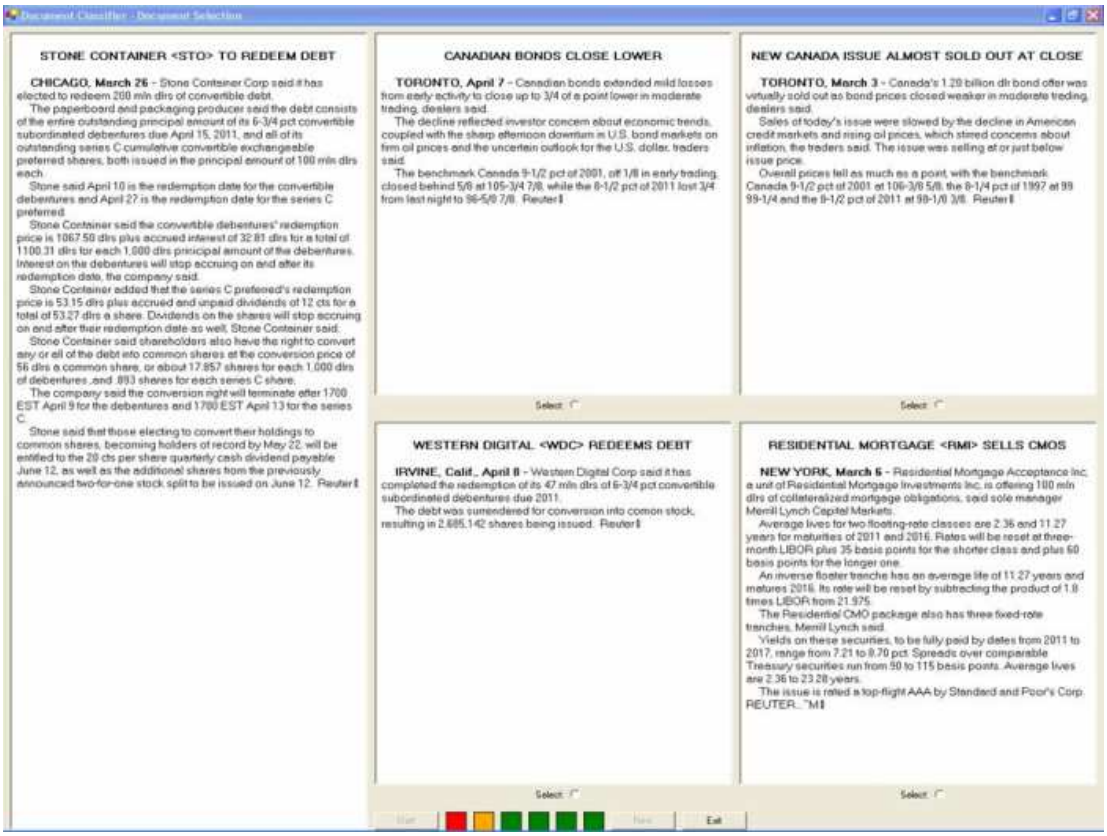

Figure 3.3: Screenshot of interface for the user trial. The single window on the left is the target to be found, the four options on the right are the available user choices. The progress bar at the bottom illustrates that this is the second iteration of feedback

the next display. There was also a progress bar showing the number of completed and remaining iterations. Since most of the subjects were unfamiliar with the specifics of the feedback algorithms, they were not told to base their decisions on textual criteria (i.e., the presence of words) but were free to make their judgment on any basis they deemed useful. A screenshot of the user interface is provided in Figure 3.3.

The target and initial display were selected from the simulated user trees in which the target was known to be present in at least one branch of the tree and the target was not present in the initial display. Every user session was thus a walk through one of the previously analysed trees. The trees were constructed on the Reuters-21578 corpus, the articles that were displayed (the targets and the given choices) were all news reports loosely connected to financial matters. Since the topics of such documents were going to be largely unfamiliar to the subjects, the task was made *interesting* by pointing out to the user that there existed at least one sequence of actions that led to the target and they had to find one such sequence for each target. This made the trial a sort of ‘game’, hopefully maintaining user interest throughout the trial.

The trial consisted of each user being given six targets one after the other, corresponding

Algorithm	Number Found	Average Scroll Rank	Average Scroll Rank Found	Average RF Rank Found	Average Time for Successful (in secs.)	Average Time for Unsuccessful (in secs.)
Rocchio Top-D	11	47.33	18.27	14.81	162	154
Rocchio Sampled	6	34	22.67	11.83	173	135.33
RSJ Top-D	7	107	77.28	12.57	105	258.2
RSJ Sampled	0	375	N/A	N/A	N/A	156.58
Bayesian Top-D	11	23.23	19.63	18.09	203.27	295
Bayesian Sampled	9	34.42	25.55	11.22	167.55	69

Table 3.6: Summary of User Trial Results

to the 3 RF algorithms and 2 display strategies, the order of which was chosen at random. The results are presented in Table 3.6. The second column titled “Number Found” gives the number of users, out of twelve, who found the target for this combination. Each target has a corresponding scroll rank (from the tree) and the “Average Scroll Rank” is the mean scroll rank of the targets chosen to be presented to the user, while the next column provides the scroll ranks of targets that the users found by the interactive process. In each successful tree, the target could potentially be present in a number of nodes of the tree. The real users who found the target each trace one of these paths. The average RF rank of these successful searches is given in the fifth column. Time estimates for the successful and unsuccessful users are given in the last two columns.

How do these results compare with the earlier results (Tables 3.1-3.4)? It is easy to see that the number of users finding the target using the sampled scheme was less than those using the Top-D scheme. This is to be expected since Table 3.1 indicates that the percentage of paths containing the target is much lower for the sampled display. In the extreme case, the RSJ algorithm using sampled display had only 0.087% of paths in successful trees leading to the target - none of the real users using this combination found the target. There is also indication of dependence of the success of the user on the time spent, unsuccessful users spent a lesser amount of time on the task.

Comparison of the three algorithms using only the data from the simulations did not reveal a clear winner. In the user trial, both the Rocchio and the Bayesian algorithm with

the Top-D display update strategy had 11 out of 12 subjects finding the target. The targets for the user trial were chosen randomly from the subset of previously generated trees where the target was known to be present in the tree. Therefore, for these two RF algorithm / display update combinations, the users were being given an *easier* task to begin with because the successful searches in these cases were only those that had a very low R_{Scroll}. Also, even though a large proportion of the targets were found, the benefit due to RF was minimal. For Rocchio with the Top-D display for example, the average RF rank for users finding the target was 14.81, a reduction from 18.27 which was the scroll rank of targets that the users managed to find. The corresponding numbers for the Bayesian algorithm with the Top-D display are 18.09 (RF rank) and 19.63 (scroll rank).

When using the sampled display strategy for both the Rocchio and Bayesian algorithms, over a 50% reduction in search effort was observed for real users. For e.g., the scroll rank of targets found by real users was 25.55 and with the use of relevance feedback the average rank was 11.22 (last row of Table 3.6). Users of the RSJ algorithm with the Top-D display update strategy obtained even better improvements.

As described in the earlier section, 12 trained HMMs were constructed - two for each combination of RF algorithm and display strategy. The first HMM was trained on all successful paths in the corresponding trees while the second was trained on the set of shortest paths from each tree. Real users were divided into two subsets - those that were successful (i.e., found the target) for that combination and those that were not. The average probability of the sequence of actions of each action-path in each subset was calculated by following the sequence of actions through the trained HMM.

Two quantities P1 and P2 were then calculated.

$$P1 = Average\left(\frac{Prob\left(ideal \mid successful\right)}{Prob\left(average \mid successful\right)}\right)$$

and

$$P2 = Average\left(\frac{Prob\left(average \mid successful\right)}{Prob\left(average \mid unsuccessful\right)}\right)$$

where $Prob\left(ideal\right)$ is the probability when the path is mapped onto the HMM trained on shortest paths only and $Prob\left(average\right)$ is the probability calculated based on the HMM trained on all successful paths. The results are given in Table 3.7.

P1 essentially gives an estimate of how close real successful users came to achieving the upper bound as estimated by the simulations. A value higher than 1 for the Bayesian algorithm with the sampled display means that most users in the trial who found the target did so through the optimal sequence of steps. Other combinations that have high values for P1 are Rocchio/Sampled (P1=0.98) and Rocchio/Top-D(P1=0.93) which are only slightly below the value for P1 obtained by the Bayesian/Sampled combination(1.07). Since the ideal user represents the best achievable performance, an algorithm that allows real users to reach this upper limit is desirable.

Algorithm	P1	P2
Rocchio / Top-D	0.93	8.84
Rocchio / Sampled	0.98	137.79
RSJ / Top-D	0.86	154.03
RSJ / Sampled	N/A	N/A
Bayesian / Top-D	0.57	15.28
Bayesian / Sampled	1.07	71.18

Table 3.7: Behaviour of real users mapped to the statistical model

If the statistical model is interpreted as defining a prescribed sequence of actions in order to be successful for a particular algorithm-display update combination, P2 measures the odds against a real user not finding the target despite following the model. The high values here indicate the real unsuccessful users were indeed the ones that did not follow the model. It can of course be argued that since the pre-computed trees were used for the user trial, the paths followed by the real successful users would have been actions that were used to train the HMM in the first place. However, the difference in magnitude between the probabilities of the two groups indicates that there are indeed patterns in the HMM which are all the more reliable because the model was constructed after exploiting the complete range of user actions over a large number of trees. This can be verified by removing the paths of the real users from the training set of the HMM, and then calculating the probabilities - the changes in the values were found to be minimal.

3.1.7 Conclusions

The experiments described in the preceeding section examined if relevance feedback and alternative display strategies can be used to reduce the number of documents that a user of a mobile device with limited display capabilities has to examine before locating a target document. In this scenario, it is possible to construct a tree representing all possible user actions for a small number of feedback iterations. This allowed determining the performance of an “ideal user”, i.e., no real user can perform better. It is therefore possible to establish an upper limit on the performance improvement such systems can deliver. The experimental paradigm has the further advantages of (i) not requiring a real user study, which can be time consuming, and (ii) the ability to simulate very many searches, thereby facilitating statistical analysis.

Using each of three relevance feedback algorithms with a display size of four documents, 100 trees were constructed. With the greedy Top-D display strategy, analysis of the trees containing the target (i.e., the successful searches) revealed that relevance feedback with Top-D resulted in close to 50% reduction in the number of documents that a user needed to examine compared with simply performing a linear search of a ranked list calculated from the initial query. It should however be noted that this number is exaggerated because of the

presence of outliers - the reduction obtained is close to 10% without these cases.

It is unclear as to why the improvement is so low. This may be due to the experimental procedure which required a user to always select one document as relevant, even if none of the displayed documents were actually relevant. More positively, it was observed that relevance feedback almost never led to worse performance for an ideal user.

The performance of the system when using an alternative display strategy in which the displayed documents were drawn from the same underlying distribution as the current scores of documents in the database was also examined. This sampling strategy approximated a strategy in which the aim is to maximise the immediate information gain from user feedback.

Using this display strategy, the Rocchio algorithm (with no explicit feature selection) and the Bayesian algorithm (which implicitly uses all the features incorporated into the distance metric) had a larger number of successful searches. However, this large improvement may be misleading. The target is present in an extremely small fraction of the 1024 paths of the tree. Thus, while the "ideal user" is guaranteed to find the target, any deviation by real users from the "ideal" is likely to result in a failed search. RSJ's offer weight selection mechanism is known to be unstable, and coupling this with an exploratory display update strategy led to worse performance.

Generalising, it is clear that if the user's query is sufficiently accurate, then the initial rank of the target document is likely to be high and scrolling or relevance feedback with a greedy display performs almost equally well. However, if the user's initial query is poor, then scrolling is futile and relevance feedback is required - either with a display strategy that explores larger regions of the search space or a feedback algorithm that does the same.

The simulation-based framework indicated that there is little to choose between the three algorithms considered. Three combinations - Rocchio/Sampled, RSJ/Top-D and Bayesian/Sampled - provided equivalent performance. This was both in terms of the probability of finding the target across a range of intial scroll ranks and the predicted upper bound performance of the ideal user. These three combinations were again similar in terms of reducing the search effort of real users. However, amongst these three, the Bayesian algorithm with the sampled display strategy led to most real users finding the target.

A method for capturing the statistical properties of the trees built was shown in the form of training a Hidden Markov Model. This HMM is a compact probabilistic representation of all the successful "users" encountered during the tree building. It was also shown that the real users who were successful mapped more closely to this trained model than the unsuccessful users.

3.2 Relevance Feedback for Web Search

Over the last few years, one form of Information Retrieval has received more exposure than all others - that of searching the Web. The size of this collection and the fact that it includes a wide range of media types makes searching it a challenging task.

Due to the unrestricted manner in which web pages are created, there is likely to be an inconsistency in the style of the actual text in the pages - this is the reason that retrieval techniques for web search are tailored to take advantage of any available side information obtained from the structure of the web itself.

Relevance feedback is a classical IR technique where users relay their agreement with the system's evaluation of relevance back to the system, which then uses this information to provide a revised list. Even with state-of-the-art search engines, users are often dissatisfied with the returned results and have to manually alter their query. Despite the presence of this gap, web search engines of today do not provide the option for feedback. Part of this is due to the fact that users do not understand the mechanisms of the RF algorithms, and partly due to the fact that providing this judgement requires some additional effort on the users' part.

This section explores the effectiveness of relevance feedback methods in assisting the user to access a predefined target document. Exploiting the fact that though the number of user choices is large, it is still limited - the experimental paradigm of examining the entire user space is used to study this problem. It is therefore feasible to generate and study the complete space of a user's interactions and obtain the upper bound on the effectiveness of one iteration of relevance feedback. This bound represents the actions of an "ideal user" whose choices enable the system to gather the most information.

3.2.1 Evaluation Methodology

The experimental procedure to examine the effect of relevance feedback is designed to include the complete space of possible user interactions with the system within the particular scenario. Assuming a display size of 10 for web search results, this gives 2^{10} ways of choosing relevant documents from the displayed set. Each such combination can be fed back into the feedback algorithm, and the position of a known item can be noted. This position can be compared with the rank of the same item in the initial ranked list to measure the potential (dis)advantage of relevance feedback. Each branch in Figure 3.4 corresponds to using one set of relevant documents from the first level. One update of the RF algorithm causes a re-ranking of the remaining documents and the change in position of a known item can be measured. The one combination which pushed the target to have the highest position is the optimal feedback. The average rank improvement in a given tree can also be calculated.

To use the tree-building evaluation paradigm with target testing, specific query-result pairs are needed. These are referred to as definitive or navigational queries, i.e., queries which have a single HTML page as their target. Such queries represent a user that has a particular site in mind, possibly because he/she has visited it in the past. The intention behind navigational queries is therefore to reach the particular site. A list of such queries was collated from an internal study, at Microsoft Research Labs Cambridge, of relevance judgements in which real users matched short web-style queries to URLs which were the

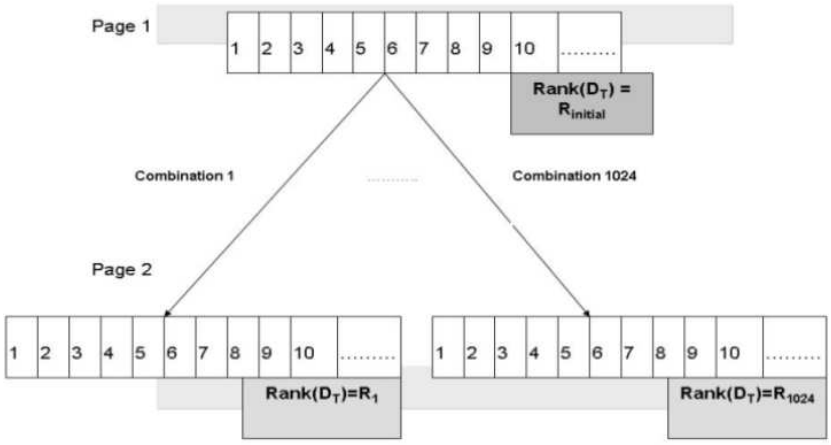

Figure 3.4: Tree for one iteration of relevance feedback, showing the rank of the target in Page 1 and each of the 1024 possible ranks resulting for Page 2 depending on the set of relevant documents chosen

answers to these queries. For every tree, the following two metrics are calculated:

$$Best\_Improvement_{tree\_n} = max(R_{Scroll} - R_i)\ for\ i = 1, 2.., 1024$$

$$Average\_Improvement_{tree\_n} = \frac{\sum_{i=1}^{1024}(R_{Scroll} - R_i)}{1024}$$

R_{Scroll} is the position of the target in the initial ranked list. Since the effect of each of the 2^10 ways of picking the feedback documents is being investigated, each combination of relevant documents results in a (potentially) different ranked list. The position of the target in the ranked list generated by using the i^{th} feedback combination is given by R_i. The difference $R_{Scroll} - R_i$ is a measure of the utility of using feedback documents set i. Ideally, this difference will be positive for all i. But in some situations, RF could result in the target document falling further down the ranking than it initially was, i.e., $(R_{Scroll} - R_i) < 0$.

The ideal user is represented by the feedback combination i which resulted in the target being being raised to the top end of the ranking. This is captured by the "Best_Improvement" metric while "Average_Improvement" measures the nett effect of all possible feedback combinations. Since the trees represent all possible user actions, an exhaustive estimate of the effects of feedback can be obtained.

3.2.2 Experimental Setup

The experiments were performed using an API that allowed querying the MSN Search Engine [MSN]. The API allowed access to the publicly available MSN search engine during the period June-August 2004. Up to 500 results were gathered for each of the navigational queries - 60% of the queries returned the result in the top 10, i.e., the first page. The subset of queries which contained the target between rank 11 and rank 500 of the returned results were used for building the trees. Over 30% of the remaining queries did not contain the target URL in the set of results returned - for many of these cases, the updated index of the search engine did not contain the target because it no longer existed. The set of initial

results returned by the search engine for each of the remaining 54 queries was used as its local database, against which relevance feedback was performed.

Evaluating how information about the relevancy of the first 10 results alters the ranking of the remaining documents is the aim of the experiments. This initial set of ten documents can be constructed in one of two ways:

1. Use the top ten as returned by the search engine
2. Use the similarity measure of the RF algorithm to re-rank the local database and then select the top 10

For the second option, the returned set of results is scored against the query - using the cosine dot product (Equation 2.2) for the Rocchio and Bayesian algorithms and BM25 (Equation 2.19) for RSJ. They are then ranked in decreasing order of scores based on this metric. This ranking would most likely differ from the ranking produced by the search engine.

This provided two ways of choosing the initial displayed set, and the effect of applying relevance feedback to each of these two rankings is investigated.

Document Representations

Two different representations of a document (the HTML page pointed to by a URL, in this case) were evaluated:

1. Up to the first 1000 words from the text of the HTML page pointed to by the URL
2. The anchor text of up to 25 other pages linking into this URL

The text from each of these two choices is extracted and term-frequency / inverse-document-frequency information is calculated to provide the representation for each document. Using only the plain text from the document makes the task similar to traditional IR.

The mechanism by which the World Wide Web evolves provides us with a wide range of features unique to this hyperlinked environment. The indegree/outdegree and URL-length are examples of such metadata. Anchor text is the visible text associated with the hyperlink from a page. The experiments are therefore testing the heuristic that anchor text is constructed in a more principled manner than the plain text of a page, and thus serves as better evidence about the contents of the page being pointed to. The usefulness of the evidence provided by anchor text for web-site finding has been shown (e.g. [CHR01]), and its use for feedback is only now being illustrated.

It is likely that search engines use a weighted combination of these two representations (and a few more). The experiments aim to measure the relative effectiveness of each representation in the context of relevance feedback.

3.2.3 Experiments and Results

The following are graphs of the results produced from the data gathered. Each point in the graphs corresponds to one query-result pair. The best improvement for that tree is plotted as the y-axis and the initial rank is on the x-axis. The average improvement can be plotted similarly. The first graph in each pair is where the search engine listing is used as the initial display set and the second graph is where the re-ranked list using the RF metric is used as the initial display.

By plotting this information, the ability of the algorithms to reach the ceiling imposed by the best case scenario is measured. The maximum possible improvement is for a document which had $10 < R_{Scroll} < 500$ to occupy the first position in the re-ranked list after feedback. Since the initial page of 10 results has been frozen, the highest rank that a document can occupy after re-ranking is 11. Hence, the upper limited is defined by the line $y = x - 11$ (the dotted line in the graphs). The closer the points in the graph approach this line, the closer their performance approaches the optimum.

In each set of graphs, the best fit straight line for the data points is also provided, as is the equation of this line. If the initial ranks and best improvements are each in arrays $\mathbf{x}$ and $\mathbf{y}$ each of size n, the best fit line $y = mx + c$ is given by

$$m = \frac{n\sum(x_i.y_i) - (\sum x_i.\sum y_i)}{n\sum x_i^2 - (\sum x_i)^2}$$

and

$$c = \frac{\sum y_i - m\sum x_i}{n}$$

In many cases, the spread of points around the best fit line indicates that the approximation provided by a linearity assumption between the X and Y axis may not be valid. To estimate the goodness of fit, each graph also has the square of the correlation coefficient (r^2) between $\mathbf{x}$ and $\mathbf{y}$ where

$$r = \frac{n\sum(x_i.y_i) - (\sum x_i.\sum y_i)}{[n\sum x_i^2 - (\sum x_i)^2][n\sum y_i^2 - (\sum y_i)^2]}$$

The correlation coefficient r lies between 0 and 1 with $r = 1$ being when the best fit line approximation is most reliable.

It is clear from Figures 3.6, 3.8 and 3.10 that the anchor text representation of documents provides the better performance for all three relevance feedback algorithms. However, the change in performance with document representation varies considerably across the three algorithms. Rocchio shows the largest variation in performance. RSJ shows the least performance improvement when using anchor text compared with Rocchio and Bayes. Most significantly, the Bayesian algorithm's performance is superior, irrespective of document representation and its performance varies least, i.e., it exhibits the least sensitivity to changes in the document representation, while exhibiting the best performance for any document representation.

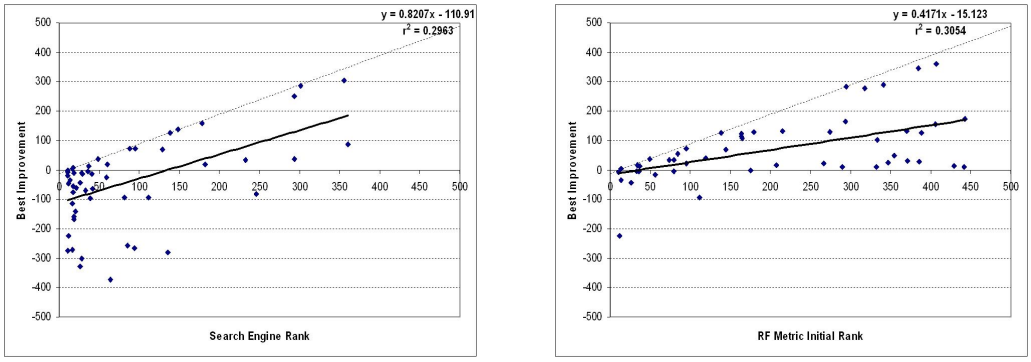

Figure 3.5: Rocchio RF algorithm with entire page as representation - distance between dotted line(optimal) and the solid line(best fit) indicates deviation from best possible

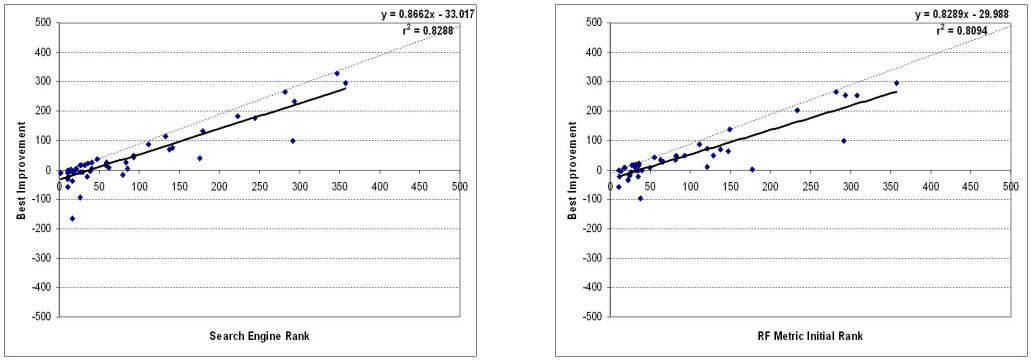

Figure 3.6: Rocchio RF algorithm with anchor text as representation some initially low-ranked documents do manage to achieve maximum possible improvement in position

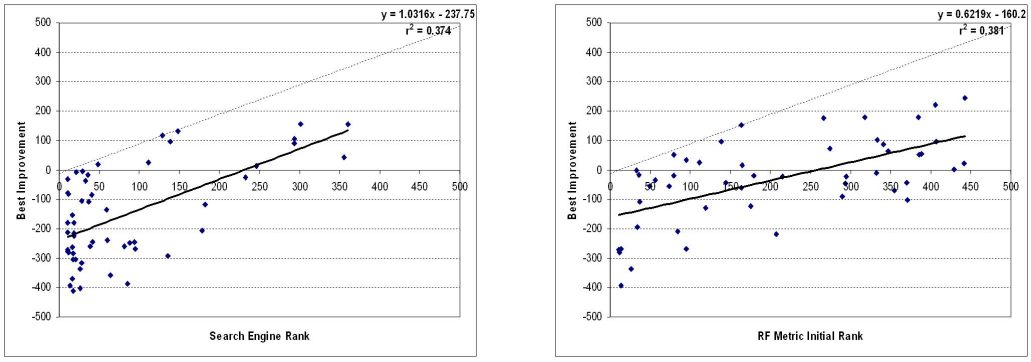

Figure 3.7: RSJ RF algorithm with entire page as representation large scatter around best fit line indicating unstable algorithm

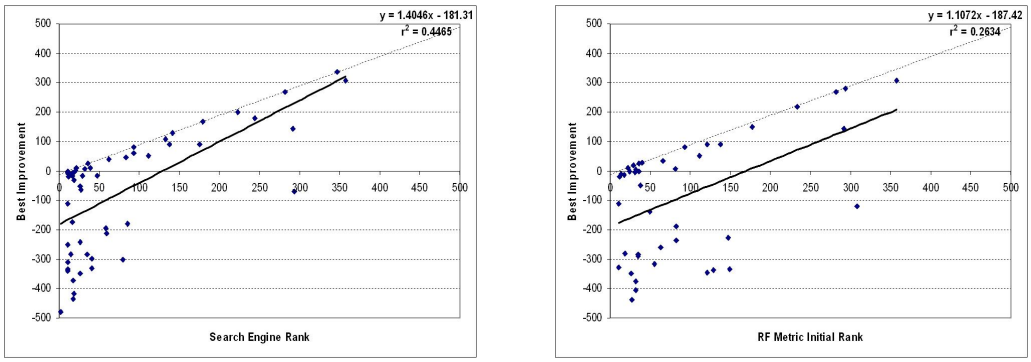

Figure 3.8: RSJ RF algorithm with anchor text as representation a few points on the optimal line, but also a number of points on the negative side of the y-axis

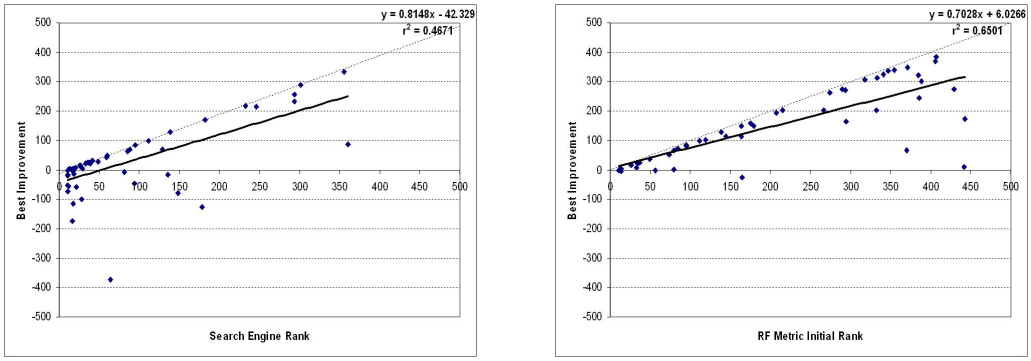

Figure 3.9: Bayesian RF algorithm with entire page as representation noise in representation affects performance

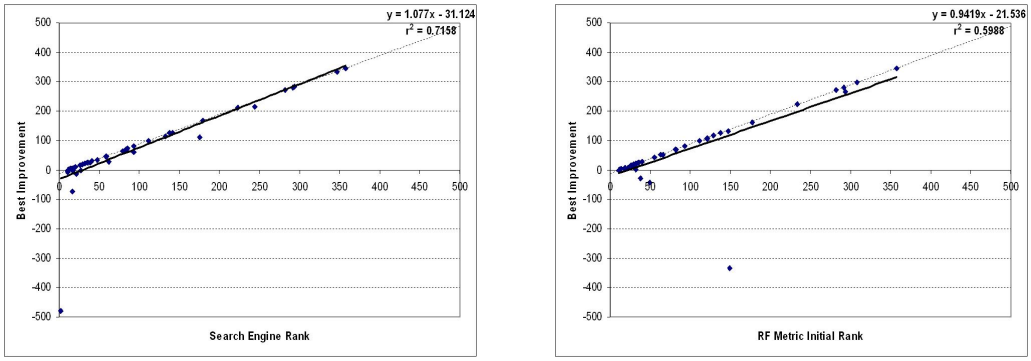

Figure 3.10: Bayesian RF algorithm with anchor text as representation tight adherence to upper bound

It is also interesting to observe that for all three RF algorithms, performance is worse when initialised using RF re-ranking compared with simply using the top 10 results returned by the search engine. Therefore, there is no evidence to suggest that having a matching between the metrics used for initial retrieval and subsequent feedback is beneficial. This would lend support to meta-search engines which could potentially leverage the capabilities of specialised individual engines for different stages of a multiple iteration retrieval session.

Figures 3.5-3.10 are for the "ideal" user, but real users' actions may be significantly different. To understand how performance degrades with deviation from the "ideal", $p_{change} = r_{change}/10$ is defined, where $r_{change} = (R_{Scroll} - R_i)$, which represents the change in the number of pages that a user must examine before locating a target document. Preferably, p_{change} should always be greater than zero, indicating that any user choice leads to a reduction in search time. Of course, this is very unlikely in practice - some user actions will surely lead to a worse ranking of the documents. However, it is undesirable to observe a situation in which few (or only one) choice leads to an improvement whilst almost all other choices were deleterious.

To investigate this, the change in document ranking, p_{change}, averaged over all user choices (1024) and over all trees was examined. Figures 3.11-3.13 depict the cumulative frequency distribution for each of the three relevance feedback algorithms. The intersection of these monotonically increasing curves with the vertical line at $p_{change} = 0$, indicates what percentage of user actions resulted in the target document being ranked worse than its initial ranking after one round of relevance feedback.

In each case, the worst combination for a particular feedback algorithm (the curve whose intersection with the vertical line is the highest) is reported. For Rocchio and RSJ, 80% of all possible user actions lead to worse performance. In contrast, for the Bayesian RF algorithm, only 60% of user actions result in worse performance.

It should be noted that these results are averaged over all possible user actions, i.e., the user actions are considered uniformly random. In practice, users will not behave randomly, and it can be expected that real users will exhibit better performance than predicted here.

3.2.4 Conclusions

The behaviour of one iteration of three relevance feedback algorithms applied to web search using two different document representations was examined. To do so, an experimental paradigm that enumerated all possible user actions was adopted. This permitted determining the performance of an "ideal user", i.e., no real user would perform better. Thus, an upper limit on the performance of each of the relevance feedback algorithms could be established.

It was observed that all three RF algorithms exhibited best performance when using anchor text as the document representation. This supports previous work ([CHR01], [KZ04]) describing the benefits of an anchor text representation for traditional web search and indicates that anchor text is also beneficial for relevance feedback.

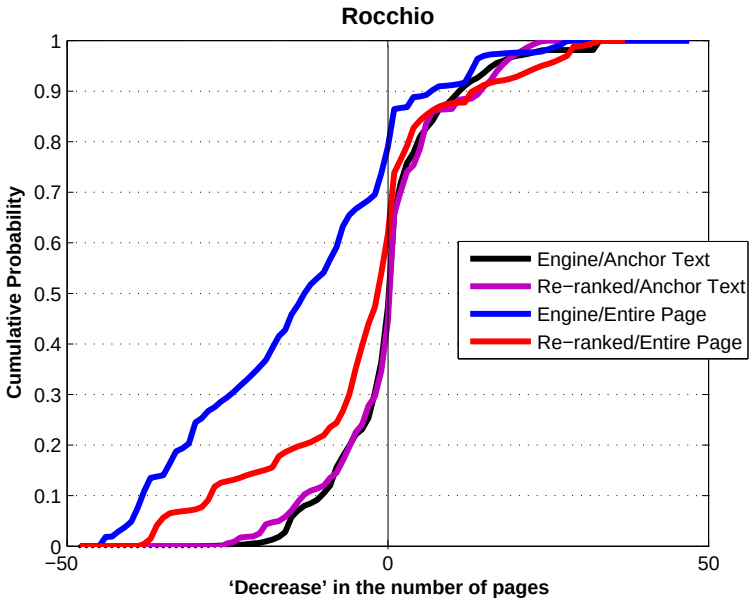

Figure 3.11: Estimating the average performance of Rocchio with both document representations

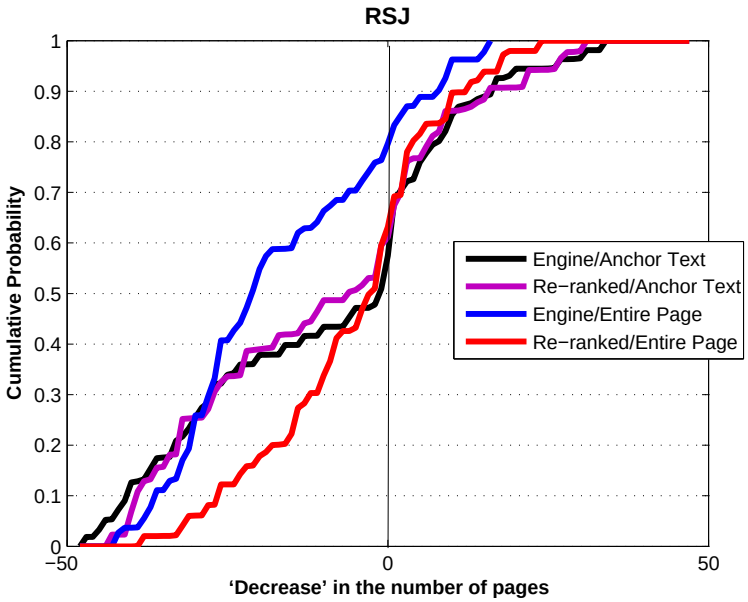

Figure 3.12: Estimating the average performance of RSJ with both document representations

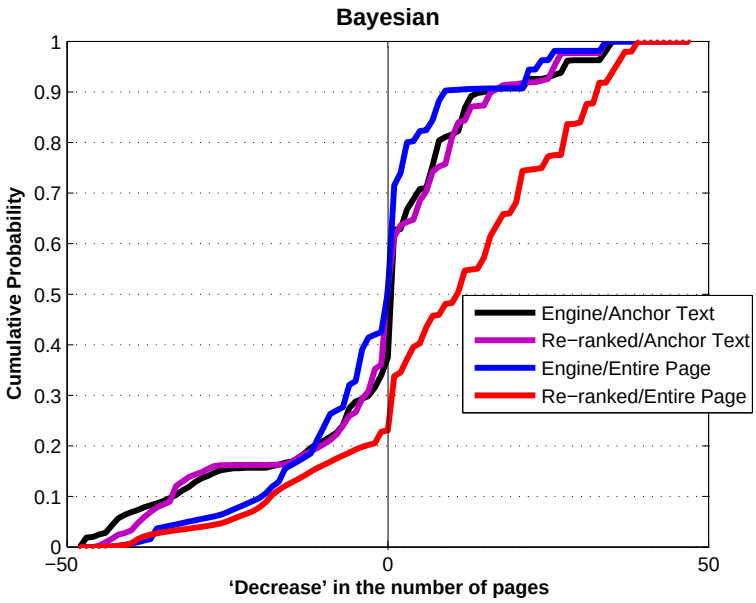

Figure 3.13: Estimating the average performance of Bayesian with both document representations

The change in performance of the RF algorithms varied considerably with changes to the document representation. The Rocchio algorithm exhibits the most variation in performance, while the RSJ algorithm showed the smallest improvement when anchor text was used. In comparison, the Bayesian algorithm outperformed both Rocchio and RSJ for both the document representations and exhibited the least variation to document representations. The performance of the "ideal user" using the Bayesian relevance feedback is in most cases almost optimal.

Interestingly, all three RF algorithms performed worse when initialised with the top-ten documents after RF re-ranking, as opposed to simply using the top-ten documents returned from the search engine. Finally, in order to investigate the effect of real users deviating from the "ideal user", the cumulative frequency distribution of changes in document rank averaged over all possible user choices and over all searches (a total of 54 for each algorithm-document combination) was examined. For Bayesian RF, 60% of random user choices lead to a worse re-ranking after one iteration of relevance feedback. This compared with 80% for Rocchio and RSJ.

For Bayesian relevance feedback, this means that if a user were to select as relevant a random set of documents from the displayed set, then 40% of the time this would lead to an improvement in ranking. This is higher than one would expect, but it is probably due to the fact that almost all of the displayed documents retrieved by the search engine are relevant to some extent. Clearly, users do not behave randomly and it should therefore be expected that the cumulative frequency distribution for real users will be even better.

A shortcoming of RSJ is its usage in collections which are homogenous. Since the retrieval for each query was performed against a local database consisting of the top 500

results returned from a reputed search engine, these results are likely to be good and possibly all similar to each other. A specific instance where this is likely to affect RSJ (more than the other algorithms) is the initial weighting of the query terms : for all algorithms, an idf based (N/n_i) initial weight was used for a query term. For homogenous collections, an initial weight of $((N - n_i)/n_i)$ would be more suitable because a large number of terms might be occurring in a majority of items in the collection.

The empirical upper bound on performance of the Bayesian relevance feedback algorithm was shown to be better than that for Rocchio or RSJ and almost approached the best possible. It was also more robust/stable to variations in the document representation. Finally, the performance for real users who deviate from the "ideal" is expected to be very good.=

Chapter 4

Descriptive Properties of Document Collections

In information retrieval (IR) it is often observed that the same set of design choices for data processing provides different effectiveness on different text collections. While some of the variation in performance can be attributed to user factors, there are algorithmic aspects that can be investigated in isolation from the user. Logically, the size of a data collection, the number of terms used in the representation of documents, the average number of relevant documents per query, the diversity amongst documents, and other related properties can be expected to affect retrieval performance. Many of these properties would also be indicative of performance of other text analysis algorithms.

Ideally, it should be possible to analyse a text collection in order to predict how well a particular algorithm will perform against it. While it appears reasonable to assume that the distribution of data points, i.e., document vectors, will affect performance, it has proven very difficult to identify a measure of this distribution that correlates with the performance of a given algorithm.

Traditionally, IR research (which includes clustering, classification and retrieval) does not address this question. Given a dataset, increasingly complex algorithms are proposed and used on the provided dataset. However, a data analysis step would provide much needed information regarding the nature of the dataset, thereby aiding in the design of the required algorithm. Apart from simply describing the data, of much more practical usefulness is the challenge of providing properties that have a correlation with actual performance. This chapter describes initial attempts towards this end.

The first property that is considered is the "clustering tendency" of a set of points. This measure reflects the presence or absence of natural groups in the data and is closely tied with the choice of representation (the document feature set and feature weighting scheme) as well as the similarity metric used. For measuring clusterability of text documents, a quantity based on the Cox-Lewis statistic (originally proposed in [CL76]) is suggested.

At the center of the design of an IR system is the choice of representation and similarity

metric. The second property described here examines what the effect of adding noise to the representation has on the stability of the behaviour of the similarity metric. While a formal and theoretical interpretation of this method is unclear, it has its roots in mechanisms for density estimation and can also be related to the clustering tendency. The following chapter provides concrete experimental evidence of the utility of this measure for the particular task of query performance prediction.

Lastly, a version of the intrinsic dimensionality of a given set of points is examined. The global dimensionality of text datasets is typically quite high. But points in a given neighbourhood (as described by the specific distance metric used) are likely to have much in common due to their similarities. This local property when averaged over the entire dataset provides an indication of the *coherence* present in the data.

In the context of the current thesis, it is hypothesised that data collections that exhibit more structure will be more suitable for the use of (pseudo) relevance feedback. Since the improvement due to RF has been observed to be different over different collections, the aim is to be able to predict this potential future benefit of using RF by examining the quantitative properties of the set of text documents.

4.1 Background and Motivation

All design choices being constant, the performance effectiveness of an algorithm routinely varies depending on the dataset being used for evaluation. A large part of this differing performance can be explained away in terms of the individual characteristics of each dataset. Amongst other things, the size of the dataset (number of documents), its dimensionality (the number of unique terms) and the relationships between the points (documents) are likely to be important factors. Even for a single text dataset, converting the collection of documents to a set of points would involve the choice of the set of features, a weighting scheme and a similarity metric. Each such choice will lead to a dataset with different characteristics. The aim of this research is to define properties of these datasets that correlates with retrieval effectiveness, thereby providing a guideline for making design choices.

At the heart of all text processing techniques is a similarity measure. For retrieval, the ranking of the documents in the collection with respect to the query is based on this measure, thereby making it an indicator of relevance. The behaviour of some similarity measures, including Euclidean distance and the dot product, is known to degrade (in terms of becoming non-discriminatory) in high dimensional spaces. This problem is commonly known as the "curse of dimensionality". It states that when dealing with very high-dimensional data, the number of data points required to sustain a given spatial density increases exponentially with the dimensionality of the input space. An alternative expression of the problem is that the number of points in unit volume decreases exponentially given a constant amount of data, with points tending to become equidistant from one another [AHK01].

For textual data, the dimensionality is equal to the number of unique terms seen across

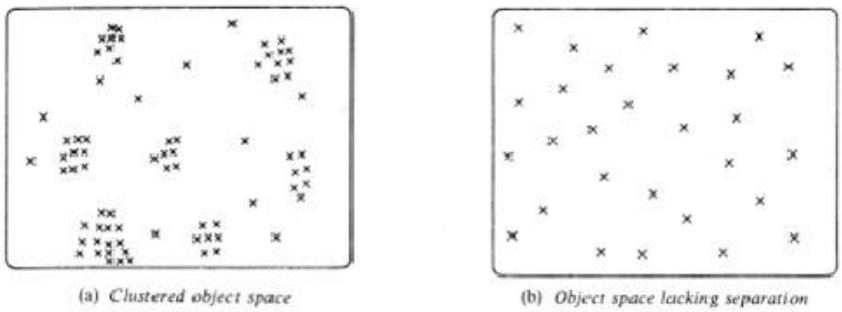

Figure 4.1: Typical object space configurations [Sal75]

the collection, typically a large number. As a consequence of the curse of dimensionality, proximity or nearness as measured by the similarity measure may not be qualitatively meaningful. And since this similarity is interpreted as relevance - any evidence gathered from one document may not provide additional evidence for other items. Further, the points (documents) becoming equidistant from each other would lead to the collection being an even distribution of points in the term space.

Though it is difficult to define what it means for a data collection to be "easy" to process, a uniformly distributed set of points is likely to qualify as being difficult because of a lack of structure that a suitably defined algorithm can take advantage of. An estimation of the uniformity of a set of points may therefore give an indication of "difficulty" for some tasks and this is the hypothesis in the current thesis.

Labelling a set of points (corresponding to documents) that are uniformly distributed as being difficult is a view mirrored by Gerard Salton in his book "Theory of Indexing" [Sal75]. As stated in the book and illustrated in Figure 4.1 (taken from the book), "when the object space configuration is similar to that shown in (a), the retrieval of a given item will lead to the retrieval of many similar items in its vicinity, thus ensuring high recall; at the same time extraneous items located at a greater distance are easy to reject, leading to high precision. On the other hand, when the indexing in use leads to an even distribution of objects across the index space, as shown in (b), the separation of relevant from non-relevant items is much harder to effect, and the retrieval results are likely to be inferior." Each point in the figure represents a document and therefore if a collection of documents represented in term space does not exhibit *structure*, it reflects a collection that is likely to provide low retrieval effectiveness.

A good starting point for some of the ideas in this section of the thesis is given in [JD88]. Section 4.6 of the book, entitled "Clustering tendency", examines the validity of the blind use of clustering algorithms on data. Most algorithms will create clusters regardless of the presence of natural clusters in the data. In order to make an informed decision as to whether

or not to commit computing resources to data clustering, it is essential to estimate the predisposition of the data to coalesce into groups. Of course, a posterior analysis could be used to establish the quality of the clustering produced by a clustering algorithm. However, it is interesting to ask whether the computational effort of applying the clustering algorithm would be justified at all. This would require the construction of a measure of the data's clustering tendency.

As mentioned above, in application areas of very large dimensionalities, such as vector space representation of data in text retrieval, certain metrics may cause all data points to become almost equidistant from each other. When the histogram of pairwise distances is plotted, a complex dataset is defined as one where there is a narrow and high peak with very light tails. Based on this intuition and some theoretical justification, Chavez and Navarro [CN01] propose the quantity $\mu^2/2\sigma^2$ as the *intrinsic dimensionality* of a set of points. Here, μ is the mean inter-point distance and σ is the variance of the histogram of distances. When points are widely separated (which is one side-effect of increased dimensionality), the mean distance between points increases. Furthermore, since every point is approximately at the same distance from every other, the variance is low. Such a dataset, with large mean and low variance for inter-point distances, has a large intrinsic dimensionality and implies uniformity that may present difficulties for applications which rely upon structure in the data representation.

There are a few problems with the direct application of intrinsic dimensionality to text retrieval. The term-document matrix representing a text collection is very sparse, i.e., contains an extremely small fraction of non-zero entries since each document may contain only a small subset of terms from the term space of the collection. Due to the sparsity of individual document vectors, the mean inter-document similarity, as measured by the inner (dot) product of vectors, is almost always equal to 0, implying that most document vectors are orthogonal to each other in the high dimensional space. Moreover, the use of the inner product as a distance/similarity measure means that this is not a metric space (the triangle inequality law does not hold). The proposed metric is based on distances and text analysis typically works in terms of similarities. One way of converting the inner product similarity measure (with an upper bound of 1 for vectors normalised to be of unit length) into a distance function is by taking *(1-simiarlity)* where $similarity$ is the cosine dot product between the document vectors. A cosine similarity value that is almost always 0 would lead to an average distance close to 1. A value of the intrinsic dimensionality calculated from such data would therefore not account for the possible structure in the set of points. Chavez and Navarro's measure might be more appropriate in situations where the data has lower sparsity and thus leads to non-zero values for inter-point distances.

Epter *et al* in [EKZ99] discuss the problem of measuring the clustering tendency of a given set of points. The authors suggest a visual method of examining the histogram of pair-

wise distances, the presence of multiple peaks in the histogram would indicate the presence of clusters. However, as has just been seen, this is likely to be ineffective in the text retrieval scenario because of minimal variance in inter-document similarity.

Another algorithm to measure the uniformity of a dataset is proposed in [SJ84]. The authors begin with the null hypothesis that the given set of M points does not come from a multidimensional uniform distribution. N additional points are sampled from such a uniform distribution and are combined with the given set of M points. A minimal spanning tree (MST) (Appendix A) over the set of $(M + N)$ points is then constructed. The number of links between the data points and the artificially generated points in the final tree is an estimate of the uniformity of the dataset - the larger the number of links, the more evidence to reject the null hypothesis. However, to build an MST requires the computation of a complete weighted graph whose nodes represent the points and the weights for the edges correspond to distances. In our case, the points would be documents, and the complexity of the algorithm would be $O(N^2)$, where N is the number of documents. This might be prohibitive for large collections.

Most relevant to the use of textual data is [EHW87]. The statistic recommended by the authors is a measurement of the density of the term-document matrix (i.e., the percentage of non-zero entries). Since every document only contains a very small fraction of all terms seen across the collection, the term-document matrix for most text collections usually contains zeros for over 99% of the entries. The term-document matrices of different collections differ in this percentage. The authors indicate that a higher density corresponds to higher clusterability. While the sparsity/density is most definitely a factor, the exact nature of this dependence is not indicated. Further, the paper only deals with a binary representation and is therefore not suitable for investigating the effect of the particular choice of representation. Of importance is the role of the geometry of the set of points, which would not only depend on the binary presence/absence of the features, but also on the weights associated with the features used in the representation.

As mentioned above, the representation of a text collection is typically very sparse. The term-document matrix is such that the entry in row j and column i provides an indication about the importance of term j to document i. Though the number of unique terms in the dataset is quite large, a particular document will contain only a small number of terms, leading to most entries in the matrix being 0. So far, the effect of the large dimensionality on the nature of the dataset has been discussed. However, comparing two datasets based on the number of unique terms in each may be misleading.

Complimentary to the idea of global dimensionality is that of a local dimensionality - documents will lie in a small given neighborhood if they are related, i.e., if they share some common terms or concepts. Certain terms will be dominant in certain neighborhoods, and it is the number of such terms (axes) averaged across a number of neighborhoods that should

be interpreted as the dimensionality of the dataset. Banks and Olszewski [BO97] suggest a sampling procedure that reflects the above argument that though the set of points as a whole inhabit a space of very large dimensionality, the locality around a particular point contains other points which all together lie on a smaller subspace. The average dimensionality of these subspaces is therefore the *local dimensionality* of a dataset.

A related idea is that of an *intrinsic dimensionality* (different from the definition of Chavez and Navarro) which indicates the number of parameters required to model the given set of points. The geometric interpretation of this quantity is that the entire dataset lies on a topological curve of dimensionality less than or equal to this value. In [FO71], Fukunaga and Olsen describe a method for calculating this quantity based on the eigenvalues of local regions of the space inhabited by the points. Also of interest is [PBJD79] which describes a method for calculating the intrinsic dimensionality using the identity of each point's nearest neighbours, and has some similarity with our peturbation analysis (described in Section 4.3).

4.2 Clustering Tendency

A set of points in a real coordinate space (e.g. documents represented using the vector space model) can display three types of spatial arrangements:

1. The points are regularly spaced (due to mutual repulsion)
2. The points are aggregated or *clustered* (due to mutual attraction)
3. The points are randomly positioned

Situations (1) and (3) above do not lend themselves to be suitable for the application of clustering algorithms because of the absence of natural grouping in the data. Different sets of points will exhibit varying degrees of tendencies to aggregate, this natural disposition to form groups is referred to as the clustering tendency.

Based on the information they use, there are different kinds of tests that measure the clustering tendency [JD88]:

- Scan tests: A fixed sized window is chosen and the given dataset is scaned for the most populous window (i.e., with the most number of points). A large large count would indicate clustering.
- Quadrat analysis: The entire space is divided into equal sized windows called quadrats. The number of points falling in each window is counted. The set of counts is known to follow a Poisson distribution under randomness.
- Interpoint distances: A clustered set of points will have an abundance of small distances, a regular set of points will have very few small distances and a random set will be somewhere in the middle. This information is used to identify the presence or absence of groups.

- Structural graphs: Unlike the other methods described above, structural graphs attempt to measure more *global* information. The test is based on the distribution of the edge lengths for a graph constructed with the given points using a suitable distance metric.

Of these approaches, all but the tests based on inter-point distances are known to be ineffective and computationally very expensive when dealing with large number of dimensions. This is certainly the case with text documents where the dimensionality is the number of unique terms (typically a very large number). Amongst the different tests of clustering tendency that are based on interpoint distances, an ideal choice for the application described here would be computationally effecient, robust with respect to characteristics of text data (e.g. possibly large number of points, high dimensionality, sparsity, etc.) and intuitive.

The statistic suggested here for measuring the clustering tendency of text documents is based on the Cox-Lewis measure, defined in [CL76]. Its multidimensional equivalent is given in [PD83]. For each of a number of randomly generated points, the distance between the randomly generated point and its closest point within the dataset (called the 'marked point') is calculated. Then, the distance between the marked point and its nearest neighbour within the given data is determined. The ratio of these two distances, averaged over a number of samples, is the Cox-Lewis statistic.

A rigorous treatment of the calculation of this statistic requires the definition of a spatial point process which models the generation of the data and also provides a null hypothesis. Points can be sampled from this distribution to serve as pseudo data points. For the Cox-Lewis test, these points are used as the initial random points. While generative models have been proposed for text documents [ML02], estimating the parameters of these models involves considerable computation. If it is assumed that two or more document collections share the same generative model, then it is possible to provide a basis for a relative comparison between the datasets that does not require normalisation.

As has been noted in previous literature [JD88], the definition of a 'sampling window' from which a random point is generated is a critical factor for the Cox-Lewis statistic. Since clustering tendency is a property that is internal to the data, the generation of the random points needs to be done with care. In particular, the data points must be sampled from within a window of appropriate size. The effect of a wrongly chosen sampling window is illustrated in Figure 4.2.

In both cases, a set of 100 points were picked uniformly at random in the interval [0, 1] in 2 dimensions - these are illustrated as crosses in the figures and represent the data-set whose clustering tendency is being measured. In case (a), the reference random point (a circle in the top right hand corner) is chosen from an unrestricted sampling window whereas in case (b), a sampling window of [0, 1] was imposed in both dimensions. In the first figure, as seen from the reference point, the data would (wrongly) appear clustered. The presence of points all around the reference point in the second case would indicate randomness.

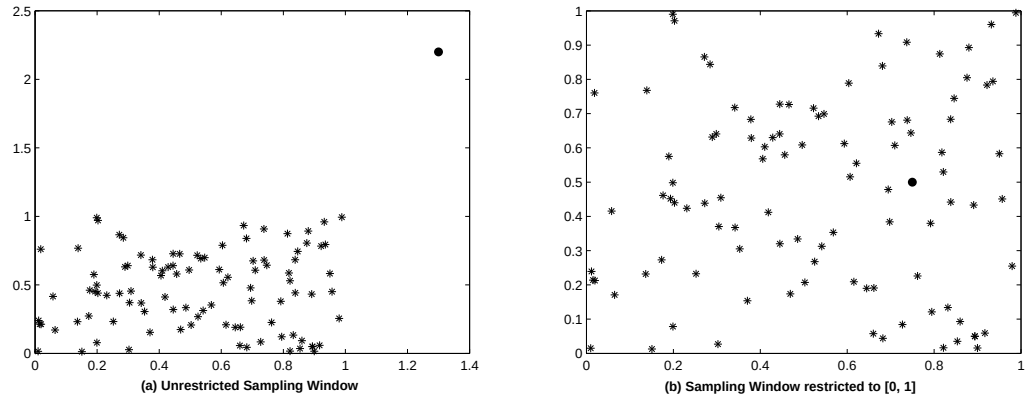

Figure 4.2: Effect of a wrongly chosen sampling window

Given a document collection as a set of points in a vector space, the size of the sampling window is defined by calculating the minimum and maximum for the vector component along each axis. This defines a hyper-rectangle. A procedure to generate a random point within this sampling window now needs to be defined. These random points (i.e., the sampling origins) serve as pseudo data points that provide a randomness hypothesis. Being substitute data points, they should share the same characteristics as the data. Of particular interest is the sparsity of the given dataset, the generated random points should on average have the same sparsity as the dataset whose clustering tendency is being measured. In order to maintain this dependence on sparsity, a point is chosen from within the dataset and its non-zero elements are replaced by a randomly chosen value along the side of the hyper-rectangle for that dimension. This provides all the details of the estimation algorithm which is described in below.

Input: *Text dataset with N documents and T terms represented using the tf-idf weighting scheme such that d_{ij} is the weight of term j in document i*

Calculate the minimum and maximum along each dimension

x_j = *Max*(d_{ij}) $\forall 1 \leq i \leq N$

y_j = *Min*(d_{ij}) $\forall 1 \leq i \leq N$

Picking a random point

For a randomly chosen document i in the collection

{

For all terms j

{

If ($d_{ij} \neq 0$)

Replace d_{ij} by a randomly chosen value between x_j and y_j

}

}

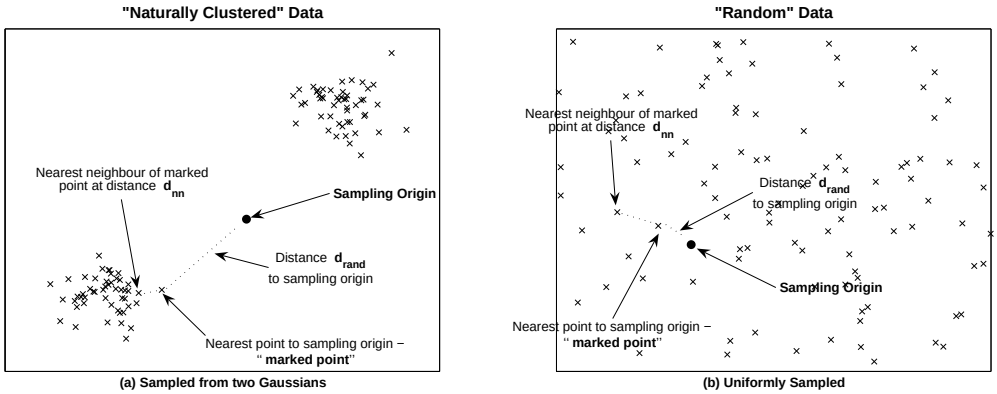

Figure 4.3: Behaviour of the Cox-Lewis statistic on clustered and random data

Calculating the statistic

Find the point p in the dataset that is closest to the random point

Similarity between p and the random point $= s_{rand}$

Find p's nearest neighbour

Similarity between p and its nearest neighbour $= s_{nn}$

Calculate $R = s_{nn} / s_{rand}$

Average R over a series of random points

Figure 4.3 gives a visual illustration of the intuition behind this method. Figure (a) shows 50 points sampled from each of two Gaussians and figure (b) shows 100 points sampled from a uniform distribution. When the data contains inherent clusters, the distance d_{rand} between the randomly sampled point and its closest neighbor in the dataset is likely to be much larger than the distance d_{nn} between the marked point and its nearest neighbour. In other words, on average the similarity s_{rand} is much smaller than s_{nn}, leading to a large ratio for s_{nn}/s_{rand}. On the other hand, data with no structure will have a smaller average ratio s_{nn}/s_{rand} and thus a larger Cox-Lewis statistic.

In the experiments described in this chapter document sets are examined using the vector space model (VSM). For calculating the Cox-Lewis statistic, the multidimensional sampling window is defined as above. Similar analysis can be performed using other IR models. For example, in the language model, the sampling origin could be obtained from the language model of the collection and the randomness statistic can be calculated by the use of an appropriate similarity metric (e.g. KL-divergence).

4.3 Perturbation Analysis

Building a text retrieval system involves making choices for the representation of the documents. This includes a weighting scheme as well as a similarity measure (which will also

be used for ranking). The ranking function part of an information retrieval system can be seen as a component that takes as input an appropriately indexed collection, and in response to a given query produces an ordered list of items. The effectiveness of the retrieval system is equated to the effectiveness of the ranking function, measured in terms of being able to reproduce the ideal ranking (for the given queries) as well as possible.

The ideal desired ranking is defined as follows and comes from the practice known as "Query by Example". The query used is a data item (here, a document) of the type the collection is made of. What can be expected as output if a document that is present in the collection is used as the query? The document used as query should be ranked at the topmost position in the ranking so produced, and any failure to do so is a failure of the system.

If the ranking function is fixed, and an ideal required output is defined as above, the stability of the system with respect to noise added at the input end can be measured. The effect of this noise, called a perturbation, on the difference between the ideal and produced rankings is then an estimate of the stability of this set of points (representing documents) with respect to the ranking function.

Consider a set of documents $\mathbf{D}$ represented using the tf-idf weighting scheme. Using the cosine similarity measure, all the elements in this set are ranked with respect to document $\mathbf{d_i}$. It should be expected that $\mathbf{d_i}$ will be ranked first with a similarity score of 1. Now, add a controlled amount of noise to $\mathbf{D}$ to produce $\mathbf{D'}$. If the similarity of all elements of $\mathbf{D'}$ with respect to $\mathbf{d_i}$ is now calculated, depending on the amount of noise added, the position $\mathbf{d_i}$ in the ranking will drop. How quickly this ranking falls is therefore a reflection of the stability of this set of points to the addition of noise. An algorithm for calculating this measure of stability is provided below.

Input:

1. *Text dataset represented using the tf-idf weighting scheme*
2. *A range for the variable α (given by α_{min} to α_{max})*

Define:

- *Variance*$(\mathbf{x}) = (\sum_{i=1}^{n}(x_i - \bar{x})^2)/(n-1)$ *where* $\bar{x} = (\sum_{i=1}^{n} x_i)/n$
- $N(m, v)$ *is a random variable sampled from a Normal distribution with mean* m *and variance* v

Algorithm:

For every term j

$v_j = Variance(d_{ij}) \; \forall \; \mathbf{d_i} \in \mathbf{C}, d_{ij} \neq 0$

For α = α_{min} *:* α_{max}

{

For all $\mathbf{d_i} \in \mathbf{C}$

{

For all $d_{ij} \neq 0$

{

d'_{ij} = d_{ij} + $N(0, \alpha * v_j)$

If $(d'_{ij} < 0)$ $d'_{ij} = 0$

}

Calculate r = *Rank of document* $\mathbf{d_i}$ *when it is used as query over* $\mathbf{C}$

}

Calculate average(r) over all documents for this α

}

Calculate slope of line of increasing rank with increasing $\log(\alpha)$

The α controls the magnitude of the added noise. When comparing two datasets, for a fixed α, the amount of noise added depends on the variance of each term v_j. If the set of points are random, they are likely to have larger variance and therefore a larger amount of noise will be added for the given α. There is therefore a relationship between this measure and the clustering tendency. A tightly clustered set of points will have a lower variance v_j for each term leading to lesser amounts of noise added for a chosen α and therefore a ranking that falls more slowly.

When a plot of rank versus α is generated for a range of values for α, it can be visually observed that there is a logarithmic dependence between the rank and α. Therefore, a plot of the rank with $\log(\alpha)$ will be a straight line and this rate of change of rank over a range of α is what is used to characterise the set of points.

4.4 Local Intrinsic Dimensionality

When given a multi-dimensional dataset the simplest way of dealing with it is to define, if possible, a model that describes the given set of points. Most statistical models contain parameters, and the smallest number of such parameters required for the modelling of the set of points is known as the *intrinsic dimensionality* of the dataset. This differs from the true dimensionality of the points which is equal to the number of features in the dataset.

In the factor analysis literature, a technique known as Principal Component Analysis (PCA) [Shl05] identifies the directions of maximum variance in the data thus identifying the principal axes that capture most of the information in the given data. In text anaylsis Latent Semantic Indexing (LSI), which uses the Singular Value Decomposition (SVD) [BZJ99] of the original data matrix, is closely related to this technique. The column space of the resulting matrix is a subspace of the original matrix and represents a “semantic” space wherein terms and documents that are closely associated are placed near one another. SVD allows the arrangement of the space to reflect the major associative patterns in the data, and ignores the smaller, less important influences. The resulting columns are interpreted as being the important “concepts” in the data.

In this context, the number of principal components in PCA (number of concepts with respect to LSI) can be seen as the intrinsic dimensionality of this set of points (documents). A known shortcoming of this approach is that since PCA (and LSI) is a global technique (it is performed on the entire term-document matrix) and involves linear transformations of the data, it ignores local patterns, which may be important in certain neighbourhoods.

As described in [FO71], a more accurate estimate of the intrinsic dimensionality can be obtained by examining the data in small local subregions. The set of points in this neighbourhood can be used to calculate the *local dimensionality* of these points. This local dimensionality can be defined in many ways. e.g.:

1. By counting the number of eigenvalues that sum to some fixed fraction (say, 80%) of the total variance
2. Use statistical techniques from machine learning to estimate this number

An example of (2) is [Min00] defined in the context of PCA. Tipping and Bishop [TB97] showed that PCA can be interpreted as maximum likelihood density estimation. Using this framework, the probability of the data for each possible dimensionality can be calculated by integrating over all the PCA parameters (the data mean and variance). Minka uses Laplace's method to approximate this integral. By integrating over all possible dimensionalities and then choosing the one providing the best fit approximation, the latent dimensionality of the data can be estimated. With the sampling strategy, based on [BO97], a point and its closest K neighbours are considered and the dimensionality of this neighbourhood is calculated. This measure averaged over many points is an estimate of the inherent dimensionality of the dataset.

For a given value of K, using the Laplace criterion, the local intrinsic dimensionality of K-neighbourhoods is calculated. If the nearest neighbours are uncorrelated (for e.g. as a consequence of the curse of dimensionality), the intrinsic dimensionality of that neighbourhood will be low. If this is true across the whole collection, it will be mirrored in a small value for the average value of the local dimensionality for this K. Typically, for small values of K, there will be some amount of correlation in the nearest neighbour (NN) set. For larger values of K, depending on the nature of the dataset, the value of the local intrinsic dimensionality will either continue increasing (larger NN sets still contain correlation) or will plateau out (lesser correlation as the size of the NN set is increased). Therefore, the rate of change of the intrinsic local dimensionality with respect to increasing size of the neighbourhoods considered provides an indication of the randomness present in the dataset.

The procedure for calculating the local intrinsic dimensionality is given below.

Input:

1. *Data that is represented in a Euclidean space (e.g. documents represented using the tf-idf weighting scheme)*
2. *A range for the number of nearest neighbours K to be considered*

Algorithm:

For every point in the collection

{

For the given range of K

{

Identify the closest K nearest neighbours using the chosen similarity measure

Estimate the local dimensionality of these data points

}

}

Calculate average local dimensionality for each given K

Output the slope of the straight line: K Vs Local Dimensionality

There are many other ways of calculating the intrinsic dimensionality, e.g. [PBJD79, BS98, Ben69, Tru76]. The particular method used here was chosen because of its applicability to PCA and therefore its relation to LSI, a technique which has been shown to have benefits in the domain of text retrieval.

4.5 Experiments

This section describes the use of each of the measures as a basis for the comparison of text datasets. Seven standard IR collections are considered. These are described in Appendix B.

All seven datasets were indexed using the set of terms obtained after the removal of standard stopwords and application of the Porter stemmer and applying the tf-idf term-weighting. The distance d_{ij} between two documents i and j was calculated as $1 - similarity$, the dot product of unit document vectors being the similarity measure.

Table 4.1 provides details of each dataset. Table 4.2 shows the results of the document perturbation experiments with α being in the range $[1, 10000]$ in multiples of 10. In this case, the slope of the line Increasing Rank Vs $\log(\alpha)$ is used as the measure of complexity - the larger the slope, the more complex the dataset. For using the local intrinsic dimensionality measure, K neighbours with K ranging from 5 to 25 in steps of 5 were considered. A small slope for the line of increasing dimensionality Vs K indicates increasing randomness as the size of the neighbourhood is increased and therefore indicates a collection of larger complexity. The results are provided in Table 4.3.

The Cox-Lewis statistic provides information about the clusterability of the datasets and singles out the Time database as being most uniform in data distribution. The seven datasets can be arranged as Time $<$ MED $<$ CACM $<$ CRAN $<$ LISA $<$ CISI $<$ NPL with NPL being the most clusterable. The Perturbation experiments provide the ranking Time $<$ MED $<$ CISI $<$ CRAN $<$ LISA $<$ CACM $<$ NPL with the NPL dataset being most sensitive to the perturbation. The local intrinsic dimensionality criterion in turn produces the ranking CISI $<$ MED $<$ CRAN $<$ NPL $<$ CACM $<$ LISA $<$ Time with CISI being the least coherent.

Dataset	Number of documents	Number of unique terms	Average document length	Number of queries
LISA	6003	12881	49	35
CACM	3204	13917	92	64
CISI	1460	8436	231	112
CRAN	1400	6437	100	225
MED	1033	9396	84	30
NPL	11429	7713	22	93
Time	424	14175	302	83

Table 4.1: Characteristics of the IR collections considered

Dataset	α				
	1	**10**	**100**	**1000**	**10000**
LISA	1.77	3.35	4.21	4.35	4.44
CISI	2.02	3.38	4.65	5.09	5.33
CACM	1.06	1.42	1.71	1.81	1.78
CRAN	1.01	1.33	1.76	1.85	1.86
MED	1.03	1.31	1.54	1.57	1.58
NPL	8.97	14.39	15.78	15.58	15.46
Time	1.00	1.04	1.19	1.22	1.24

Table 4.2: Results of perturbation experiments on the datasets

Dataset	K				
	5	**10**	**15**	**20**	**25**
LISA	3.94	8.88	13.91	18.88	23.89
CISI	3.98	8.90	13.81	18.93	23.90
CACM	3.88	8.82	13.82	18.65	23.41
CRAN	3.82	8.59	13.59	18.59	23.53
MED	3.92	8.58	13.83	18.42	23.58
NPL	3.96	8.91	13.89	18.91	23.87
Time	3.50	8.50	13.75	19.00	23.75

Table 4.3: Results of local intrinsic dimensionality experiments on the seven datasets

Dataset	% Improvement in MAP	Clustering Tendency	Document Perturbation	Local Intrinsic Dimensionality
LISA	3.00	0.58	0.28	0.998
CACM	7.64	0.60	0.36	0.997
CISI	7.59	0.42	0.08	0.977
CRAN	2.11	0.44	0.09	0.988
MED	3.20	0.36	0.06	0.983
NPL	6.21	0.80	0.61	0.996
Time	6.70	0.23	0.03	1.020

Table 4.4: Intrinsic dimensionality and clustering tendency of the four collections

Unfortunately, there is no ground truth ranking of datasets that characterises them based on their complexity.

In order to investigate if any of these measures correlate with the performance of (pseudo) relevance feedback, each individual dataset was indexed using the Lemur toolkit [LEM] after removing standard stopwords. The set of queries connected to each dataset was issued against the respective collection and all the documents in that dataset were ranked with respect to the query using tf-idf scoring.

Using the provided relevance judgements, the performance on each collection (based on mean average precision) was then calculated. For each query, the top 10 documents were then fed back into the Rocchio feedback algorithm and the reranked set of results were collected for the modified query. Mean average precision (MAP) of the modified results sets were then calculated. The results are provided in Table 4.4.

In terms of improvement in MAP, the datasets can be arranged in the order CRAN $<$ LISA $<$ NPL $<$ Time $<$ CISI $<$ CACM $<$ MED, with pRF being least benefitial for the CRAN dataset. Unfortunately, none of the properties of document collections described earlier in this chapter, were able to predict this ordering. They were therefore unsuccessful in their aim of being indicators of pRF utility on different datasets. The next chapter illustrates the utility of these measures for the task of query performance prediction.

4.6 Summary

This chapter described quantitative properties of sets of text documents that can be used to measure their *complexity*. The motivation behind the definition of such properties is to be able to identify characteristics of a text collection that can be used to predict the future performance of statistical algorithms on these datasets.

The first property, the clustering tendency, reflects the presence of absence of natural groupings within the data. Assuming that a randomly distributed set of points (documents) will be immune to most algorithms, this clustering tendency indicates if regularities are

present in the data which a suitably defined algorithm can take advantage of. To measure this clustering tendency, a metric based on the Cox-Lewis statistic was defined.

Next, a method of perturbation analysis was introduced. This property measures the stability of the similarity metric (or ranking function) towards the addition of noise (i.e., perturbations) to the input representation of the documents. The metric was constructed such that higher sensitivity to the perturbations indicated a larger degree of randomness present in the input.

The last property which was called the local intrinsic dimensionality combined two well known concepts in pattern learning literature. The intrinsic dimensionality of a dataset identifies the number of axes of the lower dimensional subspace that contains all the given data points. The local dimensionality argues that though the set of points as a whole inhabit a space of very large dimensionality, the locality around a particular point contains other points which all together lie on a smaller subspace. The proposed measure of the local intrinsic dimensionality provided an indication of the correlation present in a set of documents with lesser correlation indicating a larger complexity.

The properties were unable to pick the collections on which pseudo-relevance feedback would be most beneficial. The next chapter uses these properties for another purpose, query performance prediction. The measures described in this chapter achieve a high level of success on this task. The aim then is to be to pick individual queries on which feedback is likely to provide an increase in retrieval effectiveness.

Chapter 5

Query Performance Prediction

Search engines are designed to generate a ranking of results deemed relevant to the query. However, depending on the *quality* of each query, there is likely to be a variance in the performance on each query. This performance, measured in terms of standard metrics like Precision and Recall, can be calculated with the help of available relevance judgements in the research setting. But in the case of new incoming queries, how can the performance be estimated?

This is particularly important because an acceptable level of performance needs to be provided for each individual query. Most users have experiences of issuing a query and either finding a relevant document immediately or spending considerable time and effort to no avail. These latter searches are frustrating to users and if sufficiently frequent, a search engine risks losing users. It is therefore critical that it is understood why searches fail. And since some failure is inevitable (a search engine cannot find a relevant document if it is not indexed) it is also important to predict when such failures occur in order to take remedial action.

This chapter reviews some recent work in this direction before providing results of experiments. The properties of documents described in the previous chapter are utilised for the task of query performance prediction, going by the general rule that the less *random* the result set of a given query, the more likely it is that the retrieval has been successful. An application of this performance prediction is provided in terms of a selective application of (pseudo) relevance feedback.

5.1 Motivation and Background

There is a considerable interest within the Information Retrieval community in estimating the effectiveness of search. Having such a measure would be useful for a variety of purposes identified in [YTFCD05] and include (i) providing feedback to the user, (ii) providing feedback to the search engine, (iii) providing feedback to the database creators, and (iv) optimising information fusion for meta-search engines.

A number of strategies have recently been proposed for estimating search effectiveness.

They can be broadly categorised into two classes. The first class is based on an analysis of the query. Amati *et al* [ACR04] propose an information theoretic function in the "divergence from randomness" framework to predict the average precision for a given query. This quantity is then used to apply query expansion selectively. He and Ounis [HO04] describe a method for predicting query effectiveness based on the query length and the distribution of the inverse document frequency values for each of the constituent terms. Unfortunately, these methods were able to achieve only limited success, indicating the difficulty of the task and the need for more elaborate methodologies.

The second class of algorithms is based on an analysis of the retrieved document set. Cronen-Townsend *et al* [YTFCD05] define a clarity score that depends on the query's language model estimated from the top-ranked documents returned by the search. Independent of the queries, a collection language model can be calculated over all the documents in the corpus. The clarity score is given by the relative entropy between the query's language model and the collection language model. Queries which fit the language model of the entire document collection are considered too general, leading to a low clarity score. In contrast, a query that identifies only a subset of the collection has a high specificity and thus a high clarity score.

Yom-Tov *et al* [YTFCD05] propose a method that uses a variety of heuristic features, including the overlap of result sets obtained using the query and each of its sub-queries. It is assumed that the larger the agreement across result sets, the more likely it is that a suitable set of documents has been retrieved. Other features include the score of the highest ranking document and the number of words in the query. The features are linearly combined using weights that are estimated from a learning phase that requires a set of ranked query/response data as a training set. After training, the experimental results reported in [YTFCD05] demonstrated the best performance to date, with a Kendall-τ statistic (Appendix A) of 0.439 using the same dataset as in the work reported here.

In this chapter, four measures derived from properties described in the previous chapter, that focus on the geometry of the retrieved document set, are used for estimating query performance: (i) the clustering tendency as measured by the Cox-Lewis statistic (ii) the sensitivity to document perturbation (iii) the sensitivity to query perturbation and (iv) the local intrinsic dimensionality. Sections 5.2.1 - 5.2.4 provide a description of the applicability of each feature for this task. Section 5.3 reports experimental results for ranking 200 queries based on their search effectiveness over the TREC discs 4 and 5 dataset (Appendix B). The relevant documents for these queries are known and used to calculate the average precision for each query. The precision statistics provides the ground-truth ranking of queries that is then used for comparison with other methods and calculating the corresponding Kendall-τ statistic. Section 5.4 provides the results of some experiments where query performance prediction is used for selective application of pseudo relevance feedback. The chapter con-

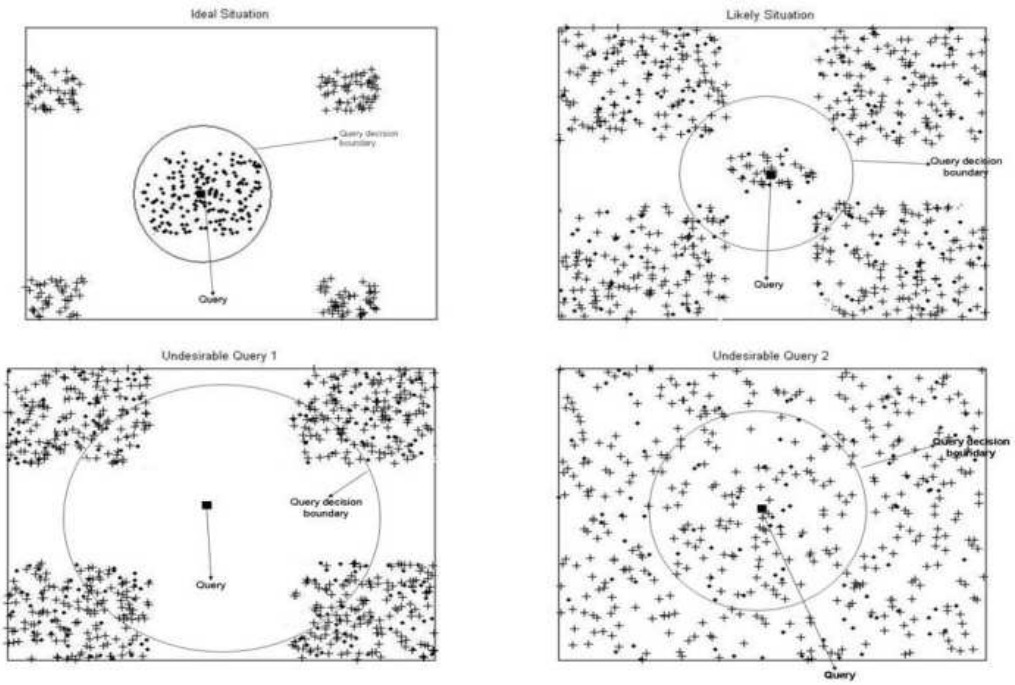

Figure 5.1: Pictures of Relevance

cludes with a summary and discussion.

5.2 Identifying the Features

Before describing the features to be used for the task, it would be useful to identify, at a very abstract level, the properties that would be desirable of a query in an information seeking environment. At this point, it is helpful to think in terms of the vector space model because the arguments that follow are based on spatial relationships.

Having made a choice regarding the weighting scheme to be used and the similarity metric, the documents can be thought of as points in a very high dimensional space. Querying this collection corresponds to generating a pseudo data point (corresponding to the query) and calculating its nearest n_S neighbours. When relevance judgments are available, these n_S neighbors are evaluated to see how closely they match the desired output (the judged relevant documents). In the case of query performance prediction, the relevance judgments are unavailable and therefore indicators which are likely to provide clues about the achieved performance need to be identified.

In the ideal case, the data point corresponding to the query would be an indistinguishable part of a larger document set which would be the exact set of documents judged to be relevant to that query. And this set as a whole would be removed from the other (implicitly irrelevant) documents of the collection. It is of course the aim of every IR system designer to achieve this situation for every query.

However, due to the particular choice of features and representation (which is more often than not based on heuristics), this ideal situation is unlikely to occur. What is more likely is that the query will lead to the retrieval of documents, a few of which are relevant and a few of which are not. In the absence of knowledge provided by an oracle, the closest neighbours (i.e., the highest scoring documents) are all assumed to relevant and their separation from the other documents in the collection can be measured.

The role of the query decision boundary is very important. This decision, in the case of range searches, would correspond to a delimiter, and the similarity of all documents that are outside it with respect to the query would be below the specified threshold. In the case of nearest neighbour queries, the decision boundary would be dictated by the number of neighbours that need to be retrieved.

There are two cases which could be deemed 'undesirable' because they do not correspond to the ideal situation. This is where (a) the result set is far from the query (b) the result set exhibits no unique structure that distinguishes it from the rest of the collection.

The first refers to the situation when the pseudo data point corresponding to the query is in a neighbourhood that is not inhabited by other documents. If a (small enough) threshold had been specified for range-based retrieval, few, if any, documents will be retrieved. Since a pre-specified number of documents are required to be retrieved in the case of nearest neighbour queries, documents which are 'far away' could also be part of the result set. Accounting for such a situation is the intuition behind using the score of the top ranking document as a clue regarding the query performance in other query performance prediction methods in literature.

When the result set contains documents that are not clusterable ('spatially random'), even though they may contain documents relevant to the query, there is no definition that makes the result set distinct from the other documents in the collection. Apart from the ideal case, there is unlikely to be a clear and well-defined boundary but rather a gradual change. However, a lack of cohesiveness amongst the retrieved documents would indicate the lack of any evidence to pick that set over any other - apart from them all individually having a high similarity with respect to the query. Measuring this cohesiveness is therefore an estimate of the quality of the result set.

Figure 5.1 provides the visual representation of the ideas in this section for the simple 2-dimensional case. In all the figures, the query is displayed as a solid square (in the center of the figure), the relevant documents are dots ('.') while the non-relevant documents are pluses ('+'). The decision boundary for the query defines the documents that will be part of the result set - the points inside the circle are amongst the nearest neighbours and the remaining points are not chosen.

It is important to note that there is no distinction made between occasions when the undesirable occurs due to a property of the query (the query is 'bad'/'difficult') as com-

pared to when it occurs due to a design choice (the representation, similarity metric, etc are unsuitable).

Recent work by Carmel *et al* [CYTDP06] provides an alternative framework for discussing the difficulty experienced when trying to retrieve relevant content from a collection of documents. The Reliable Information Access (RIA) workshop [HB04] brought together a selection of top research IR systems in order to investigate system-dependant and query-dependant factors that contribute to topic difficulty. The main finding of the workshop was that some queries are inherently difficult, and this difficulty can be expressed in the form of five distances (represented graphically in Figure 5.2) :

1. $d(Q, C)$ - The distance between the queries (Q) and the collection (C)
2. $d(Q, Q)$ - the distance amongst the queries
3. $d(R, C)$ - The distance between the relevant documents (R) and the collection
4. $d(R, R)$ - The distance amongst the relevant documents
5. $d(Q, R)$ - The distance between the queries and the relevant documents

The distance 'd' can in general be any appropriate measure (e.g. cosine). For experimental results in the paper, the authors use the Jensen-Shannon Divergence (JSD) measure given by

$$JSD(d_i, d_j) = \frac{1}{2}(D(d_i, d_j) + D(d_j, d_i))$$

where $D(d_i, d_j)$ is the KL-divergence as given in Equation 2.14.

The paper points out that the single most distinguishing quality of a *difficult query* is that it covers multiple aspects, i.e., there is variability in the different expressions of the same information need (expressed as a large distance $d(Q, Q)$) and the wide range of relevant documents that need to be retrieved to satisfy this query (a large distance $d(R, R)$). Typically in the research scenario, there is only one expression for a given query (the set Q contains only one element) and therefore $d(Q, Q)$ cannot be measured.

The paper however does show a positive correlation between $d(R, R)$ and a query's difficulty. With respect to Figure 5.1, this translates to the relevant points (the dots '.') being widely spread. However, the distance $d(Q, C)$ is shown to have minimal effect on the quality of the query, indicating that "Undesirable Query 1" does not occur in practice.

This framework for analysing queries provides clues towards features that are likely to be helpful while identifying difficult queries. The rest of this chapter describes the use of the features described in the previous chapter for the purpose of query performance prediction.

The method used in this thesis to predict query performance analyses the result sets retrieved in response to the query. Therefore, if the result set does not contain multiple aspects (where an 'aspect' refers to a topic or concept), it can be inferred that some of the aspects have been missed. A query for which the result set contains multiple aspects (i.e., a query on which the retrieval performance has been good) can therefore be identified by measuring the

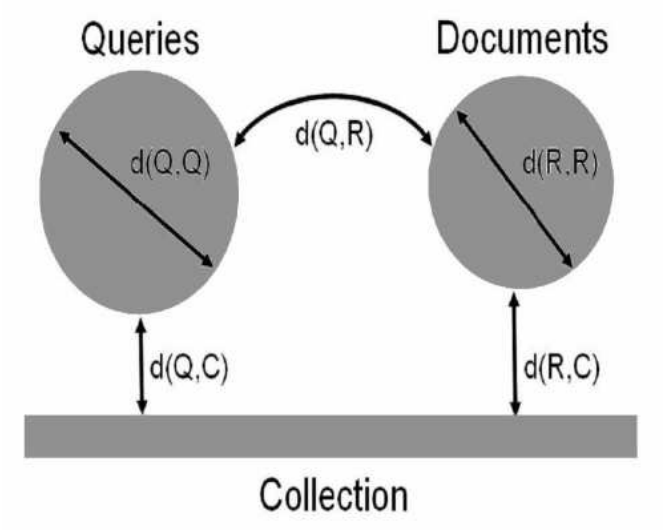

Figure 5.2: Model for explaining query difficulty [CYTDP06]

clustering tendency of the result set (the different aspects are different clusters). The modified Cox-Lewis statistic described in the preceeding chapter is used for this purpose, after a normalising factor to account for the *spread* of the aspects. In the absence of relevance judgements, the set R is approximated by the query's nearest neighbours (the result set) and the distance $d(R, R)$ here is represented by the Document Perturbation and Intrinsic Local Dimensionality features. A new feature, called Query Perturbation, is introduced to reflect $d(R, C)$.

The following sections provide the reasoning behind the applicability of each of the features for the task of query performance prediction before providing results of experiments to measure their effectiveness.

5.2.1 Clustering Tendency

The "cluster hypothesis" [vR79] states that documents relevant to a given query are likely to be similar to each other. Thus, documents relevant to a query are expected to form a group that is distinct from non-relevant documents. In practice, this hypothesis is exploited in its equivalent form: the lack of clusters in the result sets is taken to imply that the set does not contain relevant documents, provided that the set is large enough, i.e., larger than the expected number of relevant documents for the query. In other words, detecting a high level of 'randomness' in the result set implies the absence of relevant documents and thus low precision and recall for the given query.

A considerable body of literature in pattern learning covers the clustering tendency of a set of points. A good introduction to useful techniques can be found in [JD88]. For the experiments here, a modified version of the Cox-Lewis statistic [CL76] defined in Chapter 4 is used.

The Cox-Lewis statistic is based on the ratio of two distances: (i) the distance from a

randomly generated sampling point to its nearest neighbour in the dataset, called the marked point and (ii) the distance between the marked point and its nearest neighbour. A rigorous calculation of this statistic requires the definition of a spatial point process which models the generation of the data and provides the initial random points. Here, a much simplified version is used, where instead of using a spatial random process, points from within the dataset are picked and their weights are replaced by random values chosen from within a sampling window and these serve as the random generated points.

When the data contains inherent clusters, the distance d_{rand} between the random point and its marked point, i.e., the closest neighbour in the dataset, is likely to be much larger than the distance d_{nn} between the marked point and its nearest neighbor. This quantity d_{rand}/d_{nn} should therefore reflect the presence or absence of inherent groupings present in the data.

Approximation of the Cox-Lewis statistic

As has been noted in previous literature [JD88], the definition of the sampling window, a region in the data representation space from which the 'random' points are picked, is an important factor for the Cox-Lewis statistic. Since the clustering tendency is a property that is internal to the data the random points from the data set need to be chosen with care. In the experiments that follow, the sampling window is defined to be the smallest hyper-rectangle that contains all the documents in the result set.

Once the sampling window has been identified, each random point is generated by starting with a point randomly selected from the retrieved set of documents. Each non-zero term weight is replaced with a value chosen uniformly from the range that corresponds to the side of the sampling window in that dimension. The points from the dataset are thus used only to determine which components to assign a random value to.

The clustering tendency of a given set of points, here the result set, is dependant on their sparsity, i.e., the proportion of non-zero values in the matrix representation of the points. The sampling points serve as pseudo-data points and therefore replacing only the non-zero components by randomly chosen values maintains the dependency on the sparsity while eliminating the need for defining a generative model of the data.

Having obtained a random sample point, its nearest neighbour within the result set, the marked point, is determined and the distance between them is computed. The distance between the marked point and its nearest neighbour is then calculated. The ratio of these two distances provides an estimate of the randomness present within the retrieved set of documents.

When working with text documents similarity between pairs of documents or a document and a query is given by the cosine measure. For the purpose of query performance prediction, this measure is tailored further using the query-dependant extension of the dot-product described by Tombros and van Rijsbergen in [TvR04]. Amongst the alternatives

suggested, the similarity between two documents is calculated here as the product of their cosine dot product and the query-dependant component.

$$Sim_{query}\left(\mathbf{d_i}, \mathbf{d_j} \left| \mathbf{q} \right.\right) = \frac{\sum_{k=1}^{T} d_{ik} d_{jk}}{\sqrt{\sum_{k=1}^{T} d_{ik}^2 \sum_{k=1}^{T} d_{jk}^2}} * \frac{\sum_{k=1}^{T} c_k q_k}{\sqrt{\sum_{k=1}^{T} c_k^2 \sum_{k=1}^{T} q_k^2}} \tag{5.1}$$

where $\mathbf{d_i}$ and $\mathbf{d_j}$ are the two documents, d_{ik} is the weight of term k in document i, T is the number of unique terms in the collection, $\mathbf{q}$ is the query and c is the vector of terms common to both $\mathbf{d_i}$ and $\mathbf{d_j}$ with weights c_k being the average of d_{ik} and d_{jk}.

Thus, pairs of documents are close to each other if they share terms among themselves and with the query. Therefore, their distance is not an absolute value but relative to the search context, i.e., the query. If two documents do not contain query terms their query-dependant similarity will be 0 regardless of how close they may be with regards to the cosine similarity. This measure is used to determine both the marked points and their nearest neighbours in the dataset and thus obtain a query specific Cox-Lewis statistic.

As described in [CYTDP06], some queries are inherently difficult because they contain multiple aspects which can sometimes be quite different from each other. In terms of the distances used in [CYTDP06], this would correspond to a large $d(R, R)$. When using the clustering tendency as described here, the distance between different aspects will be reflected by the length of the sides of the hyper-rectangle corresponding to the sampling window. In order to reward those queries where the result set is not only clusterable (different aspects have been retrieved) but the result set contains aspects that are diverse, the computed Cox-Lewis ratio is multiplied by the average length of the sides of the hyper-rectangle.

A higher clustering tendency is thus implicated by a larger value of the ratio:

$$CT_q = Mean\left(\sum \frac{Sim_{query}\left(d_{mp}, d_{nn} \left| q \right.\right)}{Sim_{query}\left(p_{sp}, d_{mp} \left| q \right.\right)}\right) * \frac{1}{T}\sum_{i=1}^{T}(x_i - y_i) \tag{5.2}$$

where $\mathbf{q}$ is the query, p_{sp} is the sampling point, d_{mp} is the marked point, i.e., the document with largest similarity with the sampling point, and d_{nn} is the nearest neighbour of the marked point. Here x_i represents the maximum and y_i the minimum weight for a term i across the retrieved set. Both x_i and y_i are calculated when defining the sampling window. The quantity given in the above equation is taken to be proportional to the average precision of this query.

5.2.2 Document Perturbation

Given a set of retrieved documents, consider the situation in which a document is randomly selected from the result set and used as a pseudo-query over the retrieved set of documents. It should be expected that the new result list will have that very document ranked first. What would be the effect of adding noise to the representation of such a document? A perturbed version of the document is issued as a pseudo-query and the new rank that the original document assumes with respect to the search with the modified pseudo-query is recorded.

This analysis is performed for all of the documents in the result set and the rate at which the average rank changes depending on the noise is calculated. More precisely, the increase in the document rank, (as the original document falls down the list), averaged over all the documents, against the level of introduced noise is plotted. The slope of the corresponding curve is used to estimate the retrieval performance for a query.

Specifically, consider a document d_i from the result set containing n_S results for a query. Let $\mathbf{S}$ be the matrix that comprises n_S columns corresponding to the vectors of retrieved documents. When using the cosine dot product for similarities between documents, $sim(\mathbf{d_i}, \mathbf{d_i}) = \mathbf{d_i}.\mathbf{d_i^T} = 1$ where X^T denotes the transpose. This gives $argmax(\mathbf{d_i}.\mathbf{S}^T) = i$, assuming that the result set does not contain duplicate documents. Noise is added to the document vector $\mathbf{d_i}$ as follows. Every non-zero term-weight d_{ij} of this document is altered by adding to it a random value drawn from a Gaussian with 0-mean and variance $\alpha * v_j$. The value v_j is the variance for term j seen across $\mathbf{S}$. Increasing α increases the magnitude of the noise.

The perturbed document $\mathbf{d_i}'$ differs from $\mathbf{d_i}$ and therefore it will not have a similarity of 1 when compared with the unperturbed version $\mathbf{d_i}$. The number of elements in $\mathbf{S}$ that have a larger similarity with $\mathbf{d_i}'$ than $\mathbf{d_i}$ determines the new rank of the original document. As α increases, the amount of noise increases and the rank continues to fall before stabilising at $n_S/2$, which is essentially a random ranking.

It has to be noted that if a fixed amount of noise is added to a random set of points and a clustered set of points, respectively, the clustered set is likely to be more prone to a fast change in rank since the points are tightly grouped. However, the noise added here is data dependant. For a given value of α, the magnitude of introduced noise is dependant on the variance observed in the given data set. Between a random and a clustered set of points, the random set is likely to have a larger variance and therefore have a larger noise added for the same α. This, in turn, leads to a larger change in rank of the original document when the perturbed document is used as a query. It is, therefore reasonable to expect that the inverse of the rate of change of document rank with $\log(\alpha)$ will be related to the clustering properties and, consequently, to the average precision of the query.

In the experiments described here, the range of α was selected through experimentation, aiming at the amount of noise that is sufficient to induce a regular and monotonic behavior but not too high to causes erratic behavior. In order to identify the ideal values for α, a wide range was tried and a plot of ranks versus α was generated. This plot is an S-shaped Sigmoid curve, ignoring the values of α at the flat regions at the extremeties of the curve provides the range used for this dataset.

Since the perturbation process involves an element of randomness, multiple sample averaging was used to ensure the stability of the observed measure. The rank of a document at a given level of noise (α) is calculated as an average over ten samples.

5.2.3 Query Perturbation

For a specific query and a retrieval model, the terms within the query are given particular weights. Altering these weights by a controlled addition of noise produces a perturbed, "noisy", query. If the retrieval algorithm is a nearest neighbour search, the set of documents retrieved by the original query will most likely be different from the set retrieved by the perturbed query. The query perturbation feature attempts to measure how distant the original result set is from the documents in the collection that would be retrieved as a result of small perturbations in the query.

The rationale is that if the original result set forms a tight cluster that is significantly distant from other topical clusters in the collection, and if the magnitude of the added noise is small compared with the distance between clusters, then a noisy query will still retrieve most, if not all the documents from the original set. As the noise magnitude increases, the query is increasingly likely to retrieve other documents. The rate at which this occurs is used as an approximate measure of the inter-cluster distance.

This approach is similar to the document perturbation approach described in the preceding section. There, the structure of the result set was being analysed while this feature looks at the structure of the document collection in the vicinity of the query.

This measure is also related to the one described by Yom-Tov *et al* [YTFCD05] where the authors examine the overlap between the result set due to the query and the result sets corresponding to each sub-query. Sub-queries are obtained from a given query by forcing the weights of certain terms to be zero. In the method described here, the weights are perturbed by a relatively small amount. In either case, a larger degree of overlap between the results of a query and its variants indicates a more stable query whose performance is then predicted as being good.

Let $\mathbf{S}$ be a set of n_S documents retrieved in response to the query $\mathbf{q}$. Every non-zero weight q_k in $\mathbf{q}$ is perturbed by adding noise from a Gaussian of 0-mean and variance $\alpha * v_k$ to generate a new query $\mathbf{q}'$. The variance term v_k is calculated from the entire collection of documents. This perturbed query is then issued against the collection retrieving a set $\mathbf{S}'$ of n_S documents. The number of elements common to $\mathbf{S}$ and $\mathbf{S}'$ is an indicator of the query sensitivity. The overlap is calculated across a range of α, i.e., noise magnitudes.

The effect of perturbing the query is reflected in the difference between the result sets for the original and the noisy query. A comparison between these two sets can be performed using a number of measures, the simplest of which are set-overlap statistics like intersection, Jaccard's distance, etc. Since document rankings produced by queries is of interest, situations where the result sets are the same but the documents are in differing orders also need to be accounted for. To measure such differences, the edit or Levenshtein distance is used. The Levenshtein distance between two strings is the number of operations such as insertions, deletions, and substitutions, required to turn one string into the other. This distance

is used to measure the sensitivity of the result set to the perturbations of the query.

More precisely, the slope of the curves that represent the increase in the Levenshtein distance as a function of $\log(\alpha)$ as α increases is noted. This slope is expected to be inversely proportional to the average precision. Again, multiple samples (ten) were used to average possible irregular effects. The pseudo-code for the query perturbation procedure is given below.

Input:

1. *A text collection* **C** *represented using the tf-idf weighting scheme*
2. *A set of queries* **Q**

Algorithm:

Calculate $variance_j$ *which is the variance along each dimension from amongst documents in* **C** *that contain term* j

For each query in Q

{

Issue query to **C**

Collect 100 results - called the original_set

For $\alpha = \alpha_{min} : \alpha_{max}$

{

For $s = 1 : 10$

{

For each term k *present in this query*

Weight = $Original\_weight$ *+ Gaussian(0,* α******$variance_k$*)*

Issue query to the entire dataset

Collect 100 results - called the noisy_set

Find the Levenshtein distance between original_set and noisy_set

}

Find average distance over multiple samples for this alpha

}

Find average distance for each given α

Plot average distance Vs α *for the range of* α*'s*

Find slope of the line for this query

}

5.2.4 Intrinsic Local Dimensionality

In the vector space model, documents are considered to be points in a high dimensional space with coordinates corresponding to the distinct terms in the collection. However, any given document contains only a small fraction of all the terms. Therefore, while the dimensionality of the entire set of documents is high, the dimensionality of a subspace that

a sub-collection of documents occupies can be much smaller. The number of parameters required to represent a set of N points in a T dimensional space is called the intrinsic dimensionality and will always be less than $min(N, T)$.

There is a considerable literature dealing with the calculation of intrinsic dimensionality of a set of points. Fukunaga and Olsen [FO71] describe a method based on the eigenvalues of local regions in the space occupied by the points. This technique requires the definition of a threshold for significance of eigenvalues. Rather than arbitrarily fixing this threshold, the experiments described here apply Bayesian model selection using the Laplace criterion by Minka [Min00] which suggests the optimal number of components to be used for principal component analysis (PCA).

The number of dimensions required to model a given set of documents is an estimate of its "complexity". If the Laplace criterion is calculated for the whole set of documents, it gives us an estimate of the global dimensionality. Inter-document relationships, on the other hand, lead to local groupings. Measuring the number of components needed for each such restricted group gives an estimate of the local dimensionality.

Given the set of retrieved documents, for each point in this set, its closest K neighbours within the result set were identified, where K ranges from 5 to 20 in steps of 5. The number of components suggested by the Laplace criterion for this set of $K + 1$ data points, i.e., the point itself and its K neighbours, is the intrinsic dimensionality of that neighbourhood.

For a given K, the intrinsic dimensionality of the K neighbourhood of each point was calculated and averaged over the result set. As K is increased the change in the intrinsic dimensionality can be observed. The slope of the increasing intrinsic dimensionality of the result set versus K was used to predict the search performance. The underlying assumption is that a high dimensional dataset can be decomposed into a lower dimensionality component and noise. If there is a large amount of noise in the data, the number of parameters required to model this essentially random set of points is small. Therefore, the higher the intrinsic dimensionality for a given set of results, the more likely it is that the query is effective.

5.3 Experiments

The aim of the query performance prediction task is to of course be able to provide an estimate of the quality of the result set generated from a single given query. Can a mechanism be defined which can look at a user query in isolation and estimate its difficulty? This can be called 'absolute judgement' because the performance of each query is independent of all other user queries.

The experiments described in this section follow the guidelines of the TREC Robust Track 2004 and 2005 which uses 'relative judgments'. Here, the task is to provide a comparitive ordering of two queries A and B based on their individual effectiveness, i.e., amongst the two queries, which one had the better performance? When this is generalised to a list of n_Q queries, the task becomes one of predicting a ranking over the set of queries based

on the performance on each individual query. Therefore, rather than being able to predict the quality of a single query in isolation, the objective is to be able to generate a relative ordering of all the queries based on estimated performance.

The *performance* of a query can be measured in many different ways. As described in Section 2.3, a variety of measures can be used to evaluate the effectiveness of each query. As described in the guidelines of the Robust Track, the experiments here use the average precision of each query to reflect performance. Using the available relevance judgements, the average precision of each query can be calculated. The list of n_Q queries can then be ordered based on their performance, and predicting this ordering is the aim of the query performance prediction task.

TREC disks 4 and 5 were indexed using the Lemur toolkit [LEM] after removing standard stopwords. For each of the 200 TREC topics, $301-450$ and $601-650$, the description field was used to formulate a query. Two alternative retrieval methods, tf-idf and Okapi, were used to collect the top 100 results for each query. In either case, the default parameter settings were used. 100 results were considered for each query with the knowledge that the average number of assessor-judged relevant documents for this set of queries is between 60 and 70. The average precision was calculated for each query with use of the available relevance judgments. This provides a "ground truth" ranking of queries according to the average precision.

The 100 results collected in each case were converted into a set of points using the tf-idf weighting scheme where the weight d_{ij} for a term j in document $\mathbf{d_i}$ was given by $log((N+1)/(n_j+0.5)) * t_{ij}$. For each query the values of all four measures described in the previous sections were calculated for the corresponding result set. The 200 queries were ranked according to each measure and this was compared with the ground truth query ranking. The correlation (as measured by the Kendall-τ) between the average precision ranking and the ranking based on each measure is an indication of the utility of the measure for query performance prediction. Table 5.1 provides the Kendall-τ correlations between the predicted and actual ranking for each of four features.

As can be seen, for tf-idf retrieval, the document perturbation method provides the best performance with a Kendall-τ of 0.521. This compares favorably with [YTFCD05] which achieves a score of 0.439 on the same database. Moreover, unlike [YTFCD05], the method described here does not require any learning and assumes only a monotonic relationship between the features and the average precision. The other three features also perform well when compared to methods used for the same purpose, such as the use of the standard deviation of IDF of query terms, score of top ranking documents, etc. [YTFCD05, HO04, CTZC02].

The results for Okapi are slightly different. The document perturbation method continues to be the most accurate predictor of query performance with a Kendall-τ of 0.343.

Kendall-τ	Clustering Tendency	Document Perturbation	Query Perturbation	Laplace
tf-idf	0.445	0.521	0.174	0.267
Okapi	0.165	0.343	0.305	-0.050

Table 5.1: Correlation between each of the features and the Average Precision

Compared to tf-idf retrieval, the query perturbation method has a much higher degree of correlation with average precision (a value of 0.305) whereas the local intrinsic dimensionality has a negative correlation.

Combining predictive measures

Since each of the four predictive measures captures a different property of the result set, combining them could yield a further improved predictive performance. This could be achieved by constructing a problem of learning this ranking over the set of queries by considering labeled examples of pair-wise ordering between queries. In order to avoid the cost of learning, only a simple arithmetic mean of the four measures was considered. However, since each measure has values from different numerical ranges, they need to be normalised before averaging.

Three forms of normalisation were tried:

1. The same-mean normalisation. Fix one of the measures (e.g. sensitivity to document perturbation) and alter the values for the other three measures so that they all have the same mean. If the measure Y is fixed, then all values x of the measure X are changed as $x = (x \cdot mean\,(Y))\,/mean\,(X)$.

2. Min-max normalisation. Normalise each measure independently by mapping its value onto the $[0, 1]$ interval. This can be achieved by the mapping: $x = (x - min\,(X))\,/\,(max\,(X) - min\,(X))$, for all values x of the measure X.

3. Inverse-tan (arctan) normalisation. Since three of the measures: the document and query perturbation and the change in intrinsic dimensionality represent slopes, their value is normalised by applying the inverse tan (arctan) function and dividing by $\frac{\pi}{2}$. This provides the mapping onto the $[0, 1]$ interval. The values for the Cox-Lewis statistic were normalised using the min-max method in 2).

Table 5.2 shows the performance of the combined predictor for each of the three described normalisation approaches. It shows the Kendall-τ correlation between the query ranking based on the arithmetic mean, i.e., the average score of the four normalised measures, and the average precision ranking. Since any subset of the four measures can be used for prediction, the optimal combination was also investigated and the Kendall-τ for the best achieved correlation with the average precision ranking is also provided. The normalisation does not affect the performance of the individual measures since all three normalisation methods are monotonic transformations of the original scores. Thus, the performance of in-

Normalisation	tf-idf		Okapi	
	Average	Best Achieved	Average	Best Achieved
Same mean	0.561	0.561 (Average of all)	0.299	0.365 (Document perturbation + Query perturbation)
Min-Max	0.457	0.550 (Clustering tendency + Document perturbation)	0.310	0.369 (Document perturbation + Query perturbation)
Inverse-tan	0.561	0.562 (Clustering tendency + Document perturbation + Query perturbation)	0.229	0.367 (Document perturbation+ Query perturbation)

Table 5.2: Combining four search effectiveness measures

dividual normalised measure is the same as shown in Table 5.1. As expected, a combination of the features is able to achieve better performance than any feature independently.

If the two ranked lists, the actual and the predicted rankings of n entries are considered to be independent, the Kendall-τ can be approximated as a normal variable of zero mean and variance $2(2n+5)/(9n(n-1))$ (for $n = 200$, the variance is 0.0023). This means that the values for the correlation reported above are significant even at the 99.9% confidence level.

Characterising queries

One important application of methods for search performance prediction is to flag queries for which the system has not retrieved good search results before the results are presented to the user. The ability of the measures described above to distinguish successful from unsuccessful query searches is therefore explored next. The respective best performing predictors for each retrieval method (average inverse-tan normalised scores of the clustering tendency and document perturbation for tf-idf and average min-max normalised scores of document perturbation and query perturbation for Okapi) was assessed for this purpose.

The 200 queries were sorted in ascending order of the corresponding average precisions and the queries that fall into the 10%, 20%, 30%, etc., of worst performing queries according to average precision were considered. The queries were ranked according to the best performing search effectiveness measure in each case and the bottom 10%, 20%, 30%, etc., performing queries were identify. The overlap between the sets of these queries was computed to identify the agreement level. The results provided in Table 5.3 show that for tf-idf retrieval, the method can identify unsuccessful searches with a success rate between 55% and 74%. As can be expected from the lower correlation with average precision, for Okapi, the best performing predictor was able to identify unsuccessful searches with success rate

% Correctly identified	20 worst	40 worst	60 worst	80 worst
tf-idf	55	65	68	74
Okapi	25	45	58	66
Random	10	20	30	40

Table 5.3: Effectiveness of identifying the poorly performing topics

between 25% and 66%.

More often than not, the queries for which sufficient relevant content has been returned do not require any further attention. Being able to identify the poorly performing queries provide the opportunity to take remedial action, which could involve invoking alternative retrieval strategies. As illustrated in Table reftable:IdentifyWorst, the query performance measures proposed in this chapter are able to pick out the difficult queries with a success rate much higher than random thereby providing a mechanism to be able to handle such queries with extra care.

Relaxing Kendall-τ

The Kendall-τ is a non-parametric measure that indicates the correlation between two variables. It lies in the range $[-1, +1]$ with $+1$ signifying perfect correlation. Here, the two lists are the actual and predicted rankings. As described in [Voo03], the Kendall-τ might not necessarily be the most appropriate measure when evaluating a query performance prediction method. This is because it is a strict metric that penalises any differences in the lists of the two variables.

Consider two queries which have average precisions of 0.01 and 0.011 respectively. A strict ranking based on effectiveness will show that the second query has a higher effectiveness than the first one. However, for the purposes of query performance prediction, to some extent, it does not matter if they are ranked one way or the other.

This section provides a relaxed version of the Kendall-τ metric and uses it to measure the accuracy of the query performance predictor. It is suggested that this relaxed metric is more appropriate for the query performance prediction task especially when the focus is on identifying poorly performing topics.

As defined in Appendix A, the Kendall-τ metric is defined as follows. Given two lists θ_1 and θ_2 each of length n, for each pair (i, j); $i, j < n$; if i and j are in the same order in θ_1 and θ_2, then there is a penalty of 0. However, if they are in opposite order in the two rankings, then there is a penalty of 1. Therefore a penalty is added if i and j are in different order in the two lists, even if they differ by just one rank. This requirement is relaxed by only adding a penalty if i and j are in opposite orders and they differ by more than δ in both rankings. By increasing δ, a more and more lenient form of the Kendall-τ metric can be obtained. Having $\delta = 0$ gives the original *hard* version of the Kendall-τ metric. Table 5.4 provides the result for each of the four features and tf-idf retrieval with this relaxed version

	Clustering Tendency	**Document Perturbation**	**Query Perturbation**	**Laplace**
Hard Kendall-τ	0.445	0.521	0.174	0.267
$\delta = 1$	0.445	0.521	0.174	0.267
$\delta = 2$	0.445	0.521	0.174	0.267
$\delta = 3$	0.445	0.522	0.174	0.267
$\delta = 4$	0.446	0.522	0.174	0.268
$\delta = 5$	0.447	0.524	0.176	0.268
$\delta = 6$	0.449	0.525	0.177	0.269
$\delta = 7$	0.451	0.527	0.178	0.271
$\delta = 8$	0.452	0.530	0.179	0.273
$\delta = 9$	0.455	0.532	0.181	0.274
$\delta = 10$	0.457	0.534	0.183	0.276

Table 5.4: Correlation between each of the features and the Average Precision measured by the relaxed Kendall-τ

of the correlation metric.

As expected, the value of the relaxed Kendall-τ increases with larger values of δ. But importantly, the metric provides a mechanism that controls how strict the comparison between the actual (based on average precision) and predicted (based on the proposed query performance prediction measure) ranking needs to be. By increasing the value of δ, a more and more lenient version of the Kendall-τ metric can be obtained. A similar correlation measure can be used when evaluating the query performance prediction measures with Okapi retrieval.

It can be seen from Table 5.4 that the value of the relaxed Kendall-τ changes very slowly with $delta$. The objective of defining this statistic was to provide an alternative to the standard Kendall-τ which was expected to be an unreliable metric for evaluating query performance prediction methods [Voo03]. However, coupled with the results in Table 5.3, it can be seen that Kendall-τ is a suitable metric for this task.

5.4 Relation to Relevance Feedback

One of the primary objectives of developing good methodologies to predict query performance is to be able to invoke strategies for each of the query *types*. An indication of the average precision of a given query is therefore useful to flag the application of specific alternatives. The strategy that is of interest in this thesis is relevance feedback. It has been shown [ACR04] that query expansion does not provide an improvement for queries with very low and very high average precisions. The next set of experiments investigate if this is true for the case of pseudo relevance feedback (pRF) [CH79]. As described in Chapter

2, this is a practice where documents ranked high in an initial retrieval are assumed to be relevant. The system uses the top ranking documents as positive examples without the user explicitly labelling them. The re-ranked list produced by the RF algorithm is used to pick the first display set presented to the user.

Rather than using pRF indiscriminately over all queries in a set, the aim is to apply it selectively to a few chosen queries. Using each of the features described earlier, the average precision of a given query can be predicted and thus the *good* queries can be separated from the *bad*.

As before, each query from the set of 200 is used to retrieve 100 documents from the indexed collection. For tf-idf retrieval, the clustering tendency (as measured by the modified Cox-Lewis statistic) and the sensitivity to document and query perturbation of each result set was calculated. Using the min-max normalisation for the clustering tendency and arctan normalisation for the other two features, each feature was given a value in the $[0, 1]$ range. A threshold for the value of the features was used to decide which queries would be suitable for relevance feedback. The ideal value for the threshold was obtained using a sweeping exhaustive search between 0 and 1. Using the two features that individually had the highest effectiveness for query performance prediction and the best performing predictor, the queries on which to use pRF were identified.

Similarly, for Okapi retrieval, the document perturbation and query pertubation measures (i.e., the two most successful individual predictors) and their combination using the min-max normalisation (i.e., the best predictor) were used to identify optimal thresholds for picking queries which would be ideal candidates for pseudo-RF.

As a baseline for comparison, there are two values. The first is where feedback was not used for any of the queries and the mean average precision (MAP) over the entire set of 200 queries was calculated. For the second alternative, pRF was used for all the 200 queries, using the top 10 documents as the feedback candidates for each query. 100 documents were retrieved after feedback for each query and the MAP for the query set was calculated. Selective application of pRF for tf-idf was done based on (i) the clustering tendency (ii) document perturbation (iii) the best estimator (clustering tendency + document perturbation + query perturbation). In the case of Okapi retrieval, the measures used for selective pRF were (i) document perturbation (ii) query perturbation (iii) the best estimator (document perturbation + query perturbation). The results are provided in Tables 5.5 and 5.6.

The threshold provides a decision rule to pick the queries on which pRF should be used. The rule could be such that only queries for which the value of the performance prediction measure is above the threshold are used for feedback. When the threshold is low, this corresponds to using all but those queries for which the performance has been predicted as being bad. An alternative decision rule is when a high threshold is used and all queries for which the prediction measure is below the threshold (i.e., all but the predicted

MAP before PRF	0.1469
MAP after PRF	0.1537
Selective PRF with clustering tendency	0.1562 (for results sets with value below threshold 0.920)
Selective PRF with document perturbation	0.1592 (for results sets with value below threshold 0.807)
Selective PRF with the best predictor	0.1554 (for results sets with value above threshold of 0.055 for the normalised mean)

Table 5.5: Selective use of pseudo Relevance Feedback with tf-idf retrieval

MAP before PRF	0.1882
MAP after PRF	0.2080
Selective PRF with document perturbation	0.2094 (for results sets with value below threshold 0.852)
Selective PRF with query perturbation	0.2099 (for results sets with value below threshold 0.99)
Selective PRF with the best predictor	0.2083 (for results sets with value above threshold of 0.034 for the normalised mean)

Table 5.6: Selective use of pseudo Relevance Feedback with Okapi retrieval

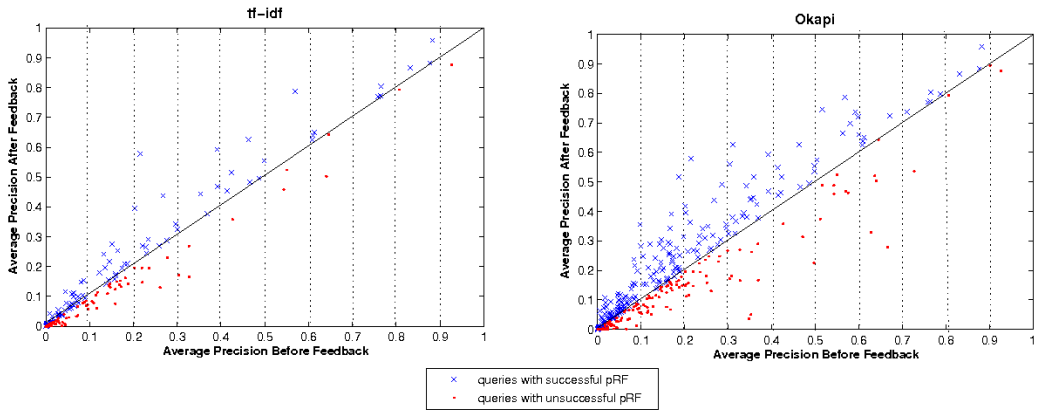

Figure 5.3: Pseudo Relevance Feedback

best performing queries) are used for pRF. A range in the middle can also be used. All the above alternatives were tried.

As can be seen in the table, there is some minimal benefit in applying pRF selectively. With tf-idf retrieval, when the clustering tendency measure is used as the basis for making the decision, the largest improvement is obtained when using pRF only for queries for which result sets have a value of clustering tendency less than 0.920 and for document perturbation the threshold was 0.807. In both cases, the queries chosen for pRF are all but those that are predicted to be the best performing ones. In case of the best predictor, the decision rule is to use all but the worst performing queries for pRF. The behaviour of all the features are therefore in line with the results shown in [ACR04]. Similar levels of improvement were observed when using pRF with Okapi retrieval. For both the document perturbation and query perturbation measures, queries with the value of each measure below a high threshold (0.852 and 0.99 respectively) were suggested for feedback. For the combined measure, all but the worst performing queries being used for pRF led to an extremely minimal improvement over indiscriminately apply pRF over all queries.

The disappointing observation however is that in all cases, the improvement due to selective application of pRF is minimal (when compared to using pRF over all queries). In order to investigate why this is so, a plot of the average precision of each individual query before and after pRF was generated (refer Figure 5.3). The queries for which the average precision increased after feedback are plotted as blue crosses while the remaining are red circles. The diagonal line separates those cases for which pRF should have been applied (i.e., those that lead to an increase in average precision) from those that should not have been used. The X-axis is divided into cells corresponding to intervals of 0.1 for the average precision. Identifying the set of queries which would have been ideal candidates for pRF based on the initial average precision corresponds to picking the cell in which the blue crosses above the line outnumber the red circles below.

As can be seen from the figure, the only intervals which show this behaviour for tf-idf

retrieval is for initial average precision in the range $[0.4, 0.5]$ and $[0.7, 0.9]$. No such interval exists for the case of Okapi retrieval. This indicates that even if complete information regarding the average precision of a given query was available, it would still not have been possible to decide based on only this information if pRF would be beneficial or not. The query performance prediction measures were only substitutes for the average precision and therefore were not very useful in this task.

5.5 Conclusions

In this chapter results on estimating search effectiveness by examining properties of the result set and the documents in its vicinity were presented. The starting hypothesis was that an effective search will result in a result set that exhibits structure since relevant documents are likely to be similar and cluster together.

Four measures were investigated: the clustering tendency, the sensitivity to document perturbation and query perturbation, respectively, and the rate of change in the local intrinsic dimensionality. Three of these measures are focused on examining the original set of retrieved documents while the query perturbation sensitivity examines the structure of the document collection in the vicinity of the query.

Experimental results with TREC disks 4 and 5 and topic sets 301-450 and 601-650 show a considerable improvement over the previous attempts to predict search effectiveness when using tf-idf retrieval. It was demonstrated that by considering the sensitivity of the result set to document perturbation a Kendall-τ correlation of 0.521 can be achieved with the average precision ranking of queries. Combining this measure with the clustering tendency, based on the Cox-Lewis statistic, and the query perturbation measure leads to further improvement. It should also be noted that every combination of the different measures was able to achieve a better effectiveness than considering any single individual measure. A Kendall-τ correlation of 0.562 was obtained with the average precision ranking by using all but the Laplace measure. These results are higher than previously reported results for the same document and query collections. Also, the best performing method can correctly detect 65% of the worst 20% searches. This is achieved without a significant computational cost by considering only 100 documents per query.

The measures showed a reduced ability to predict query performance when the result sets were obtained with the Okapi model. Again, the document perturbation measure achieved highest correlation with average precision (a Kendall-τ of 0.343). When combined with the query perturbation measure, a Kendall-τ of 0.369 was obtained with an average precision ranking of the queries. Consequently, for this model of retrieval, only 45% of the worst 20% searches were identified.

Despite the lower performance with Okapi, the experimental results support the assumption that the lack of structure implies a low search effectiveness. Importantly, the clustering tendency and document perturbation features only require computations based on the re-

trieved document set. It is unclear as to whether or not the high degree of success achieved on this dataset transfers to other situations, the encouraging results on the standard set of documents and queries used in the IR research community for this task lends credibility to the proposed measures.

Since this is an active research area, a number of different measures achieving varying degrees of success are avaiable for this task. Each of these measures attempt to characterise different aspects of a difficult query. It would therefore be logical to expect that a combination of these measures using machine learning based algorithms will be able to achieve a higher degree of success than any single measure. Designing and developing such methods that are able to learn to estimate query difficulty is an interesting future research area.

A pre-retrieval estimate of query effectiveness would be most desirable. However, in view of the comparatively low performance of such techniques, it is to be expected that an optimal approach will involve analysing the result set but with no involvement of the search engine in additional processing. This is particularly important in the case of meta-search where analysing the returned results of each engine is much more feasible than repeated queries. The measures based on document perturbation and measurement of clustering tendency offer such alternatives. Of course, the features require some post-processing on the result set. However, this cost is small compared with the cost of a new retrieval.

The two individual best measures and the most accurate combination for each retrieval model were then used to identify those queries for which it might be profitable to apply pseudo-relevance feedback. It was shown that this selective application of feedback provided better performance, over a set of 200 queries, over no feedback at all or blindly applying pseudo-RF to all the queries. This improvement was however minimal.

The measures that were defined and explored are not restricted to estimating search effectiveness. They can be used for comparing the complexity of different document collections and the effects of different document representations on search.

A part of the work described in this chapter was published in a paper titled “On Ranking the Effectiveness of Searches” [VCMFW06] at SIGIR 2006.

Chapter 6

Conclusions

This dissertation explored three specific issues in the context of text retrieval. These are:

- Evaluation of relevance feedback
- Quantitative properties for describing document sets
- Query performance prediction

This final chapter summarises the experimental results presented in earlier parts of the dissertation, before considering possible future directions for work arising from this thesis.

6.1 Results Summary

Chapter 3 presented a simulation-based evaluation framework that can be used for measuring the effect of adding relevance feedback (RF) to the retrieval process. This methodology was a brute force exploration of all possible user actions thereby aiding in the identification of an empirical upper-bound on the effectiveness of RF. As part of the simulation, every alternative available to the user during an interactive information retrieval session was enumerated. By design, such an investigation contains within it the sequence of actions that any real user might follow. This method therefore reduces the need for expensive and time-consuming user-trials.

Due to the computational constraints imposed by such an approach, only interfaces offering limited interactivity can be explored. Two such scenarios (searching on small displays and web-search) were considered and experimental comparison of three standard RF algorithms (Rocchio, RSJ and Bayesian) was provided. A summary of the results is as follows:

- Multiple iteration feedback with the greedy Top-D display exhibited undesirable convergence side-effects across all three algorithms. This was due to the use of a greedy display update strategy that always picks the top ranking documents for subsequent displays after re-ranking. An alternative display strategy was suggested which probabilistically sampled from the underlying score distribution over the elements in the data collection. This display update strategy was then shown to provide better performance (in most cases) over the greedy display.

- During simulations for small display devices, there was not much to choose amongst the three algorithms. However, based on a small user trial that was conducted, the Bayesian algorithm with the sampled display update strategy was chosen as being the best.
- When considering web-search, using the text of the web-pages (which reduces the problem to conventional IR) for relevance feedback was shown as being unstable. But since the hyperlinked structure of the web provides alternative sources of evidence, alternate document representations can be investigated for the use of RF. The use of anchor text was then illustrated and across all three algorithms, an improvement in effectiveness was observed.
- Since the potential upper bound was known, the performance of each algorithm could be compared to this ideal scenario. The Bayesian algorithm with anchor text approached the best achievable performance almost all the time whereas RSJ showed maximum variance with respect to the choice of representation. It was also shown that only 40% of the choices led to a drop in effectiveness when using the Bayesian feedback algorithm over not using any RF.

Chapter 4 introduced some discussion for the need of predictive measures in IR. The chapter argued in favour of a data analysis step that analyses a text collection in order to predict how well a particular algorithm will perform against it. This could provide much needed information regarding the nature of the dataset, thereby aiding in the design of algorithms (whether it be classification, clustering or retrieval) to work on the given data. Apart from simply describing the data, of much more practical usefulness is the challenge of providing properties that have a correlation with actual performance.

Three properties were described for this purpose:

1. Clustering tendency which measured the natural tendency of the points corresponding to documents to fall into groups
2. Document perturbation analysis that is indicative of the sensitivity of the similarity metric used to noise added to the representation of documents
3. Local intrinsic dimensionality which is measured in small neighbourhoods in the data and is reflective of the coherence between every point and its nearest neighbours

It was expected that the ordering of standard IR datasets based on these measures might reflect the potential improvement in retrieval effectivness that using pRF on each of the datasets is likely to provide. The results however showed that none of the three measures were able to achieve this objective.

Chapter 5 used the three properties described in chapter 4 along with a new query perturbation measure to characterise the complexity of result sets of queries. It was argued that queries with result sets that had minimal structure correspond to those queries that were

inherently difficult for our retrieval algorithm to handle. And correspondingly, the presence of structure in the result set indicates a query that has been addressed appropriately.

Experimental evaluation of each of our four measures as query performance predictors showed a significant improvement over the previous attempts to predict search effectiveness (in terms of average precision) when using tf-idf retrieval. The results were comparitively lower when using Okapi's BM25 as the retrieval function. Previous literature indicated a dependence of the effectiveness of feedback on the quality of the initial retrieval. Experiments describing the selective use of pseudo-relevance feedback based on the best performing predictors provided a small improvement over blindly applying pseudo-RF to all the queries.

6.2 Future Directions

The work presented in this dissertation looked at specific problems arising in the use of relevance feedback. However, it points towards much that still needs to be discovered in the field of text retrieval.

The need for alternative display update strategies (described in Chapter 3) is one that has received suprisingly little attention in IR literature. The probabilistic update method illustrated the need for an interplay between exploration and exploitation, but it is to be expected that other more optimal sampling strategies exist which provide a better balance. Making use of ideas from information filtering, active learning and semi-supervised learning might point towards better methods for display updates.

The work described in this thesis dealt with quantitative properties of document sets, queries and the retrieval process and showed that these can be useful indicators of retrieval effectiveness. However, the methods and measures used here are by no means comprehensive. Logically, the size of a data collection, the number of terms used in the representation of documents, average number of relevant documents per query, diversity amongst documents, and other related properties can be expected to affect retrieval performance. To the author's knowledge, no systematic study has been conducted to evaluate the contribution of these factors to the effectiveness of retrieval techniques.

For the query performance prediction task, the effectiveness of each measure varied depending on what retrieval function was used. This indicates a possible relationship between the similarity metric and the "randomness hypothesis". Understanding the nature of this relationship and then using it to develop a general method for query performance estimation across a wide range of retrieval models would be an obvious immediate problem. Extending the measures described in Chapter 5 to hyperlinked environments (where non-textual features are used as part of a document's representation) is also an interesting avenue.

Information collections do not exist in isolation, it is therefore unreasonable to expect algorithms designed in seclusion to have success across the board over a range of scenarios. Identifying indicators that not only explain differing performance of algorithms but also directs the design of new ones is therefore of utmost importance. The author hopes that this

dissertation comes of some use in addressing these and related topics in future research on text retrieval.

Appendix A

Glossary

Ad-hoc retrieval

An information retrieval task where a ranked list of answers is returned in response to a user-input query.

Clustering Tendency

The predisposition of a set of points to group together.

Collection

A set of documents which is being searched.

Display Set

A subset of the result set of a query that is shown to the user.

Document

An individual element in a database. In this thesis, documents contain only text but in general they can contain any multimedia content. The raw form of the element (sequence of words here) and its representation are referred to as documents.

Document Perturbation

A method for analysing document sets by measuring the effect of adding noise to their representations.

Inverse Document Frequency

The fraction of the entire collection in which a given term occurs.

Kendall τ

The Kendall τ statistic is used to measure the degree of correspondence between two rankings. Given two lists θ_1 and θ_2 each of length n, for each pair (i, j); $i, j < n$; if i and j are in the same order in θ_1 and θ_2, then there is a penalty of 0. However, if they are in opposite order in the two rankings, then there is a penalty of 1. Given a list of length n, there

are $(n(n-2))/2$ possible pair-wise comparisons. If the number of pairs for which there was no penalty (the concordant pairs) was c and the number of pairs for which a penalty was added (the discordant pairs) was d, then

$$Kendall - \tau = ((c-d) * 2)/(n(n-1))$$

It has a value of $+1$ if the agreement between the two rankings is perfect and -1 if one ranking is the exact reverse of the other. For other situations, the value lies between $+1$ and -1 with larger values indicating larger agreement.

Local Intrinsic Dimensionality

The number of parameters that are required to model the set of points lying in a given neighbourhood.

Minimal Spanning Tree

Given a weighted undirected graph where the weight associated with each edge represents a *distance* between the vertices being connected, the weight of the entire graph is the sum of all its edges. A minimal spanning tree can be defined as the sub-graph (more specifically, a sub-tree) of the given graph such that all the vertices are connected and the sum of the weights of the edges is the minimum.

Precision

The fraction of documents in the result set that are relevant to the user query.

Query

The information need of the user that is an input to the information retrieval system. For text retrieval, this is usually a sequence of words.

Query Perturbation

A method to predict query performance that examines what effect the addition of noise to the query representation has on the result set.

Recall

The fraction of all relevant documents to the query that were returned by the IR system as part of the result set.

Relevance Feedback

The collective term given to a wide range of techniques where the initial query entered by the user is successively modified based on evidence on evidence that is provided to the system in each iteration.

Result Set

A list of documents that the IR systems thinks is relevant to the user query. Every element of this set is typically associated with a measure of this potential relevance.

Similarity measure

A function that takes as input (the representation of) two documents and returns a numerical score such that a higher output indicates a greater degree of correlation between the two documents.

Stemming

The process of reducing morphological variants of a word to the same root by stripping off the prefix and/or suffix information.

Target Search

A search scenario where the user's information need is satisfied by a single document.

Term

The features typically used to represent documents. They can be words as seen in the document, stemmed words or keywords identified by automatic or manual procedures.

Term-Document matrix

A matrix constructed such that the entry in row *j* and column *i* represents the degree to which term *j* characterises document *i*.

Term Frequency

The number of times a given term occurs in the document.

Appendix B

Datasets

The Reuters-21578 Collection

This freely available collection consists of documents that appeared on the Reuters newswire in 1997. Typically used for text categorisation tasks, the collection is distributed as 22 files, each consisting of up to 1000 documents. The meta-data available for each document includes Date (of creation), Topics (a list of category labels) and Author. The text part of each document consists of a Title (the headline of the story) and Body (the content) section. The collection has now been superceded by the RCV1 collection for text categorisation experiments. More information about Reuters-21578 can be found at http://www.daviddlewis.com/resources/testcollections/reuters21578/readme.txt.

The Small Collections

A set of publicly available collections that have been used in the past but are considered small for current research and have therefore been superceded by larger and more exhaustive datasets. The different collections are:

- LISA: A test collection of Library and Information Science Abstracts collected at Sheffield University with natural language queries obtained from students.
- CISI: A collection of 1460 documents
- CACM: A collection of titles and abstracts from the journal CACM.
- CRAN: The larger form of the collection with 1400 documents
- MED: A collection of articles from a medical journal.
- NPL: The NPL (also known as the VASWANI) collection is a collection of around 10,000 document titles.
- Time: A collection consists of articles from the magazine Time.

More information can be found at http://www.dcs.gla.ac.uk/idom/ir_resources/test_collections/

TREC Disks 4 & 5

As part of the National Institute of Standards and Technology (NIST), Text REtrieval Conferences (TREC) are held every year and provide a forum for discussion of the latest research

in Information Retrieval. To facilitate such a discussion, NIST provides a standard collection of text documents on which all participating teams report results. For ad-hoc retrieval, the dataset currently being used are TREC disks 4 and 5. Disk 4 includes material from the Financial Times Limited (1991, 1992, 1993, 1994), the Congressional Record of the 103^{rd} Congress (1993), and the Federal Register (1994). Disk 5 includes material from the Foreign Broadcast Information Service (1996) and the Los Angeles Times (1989, 1990). Every year NIST also releases a set of topics (queries) which are used for the experiments. NIST is also responsible for hiring assessors who map each query to a set of relevant documents in the collection thereby providing a set of relevance judgements that can be used for evaluating IR systems. More information about TREC can be found at http://trec.nist.gov/.

Appendix C

Publications

1. Vishwa Vinay, Ingemar Cox, Natasa Milic-Frayling, Ken Wood: **"Evaluating Relevance Feedback Algorithms for Searching on Small Displays"**: European Conference on Information Retrieval (ECIR) 2005, Santiago de Compostella, Spain
   Extended version appeared as Vishwa Vinay, Ingemar Cox, Natasa Milic-Frayling, Ken Wood: **"Can constrained relevance feedback and display strategies help users retrieve items on mobile devices?"** in the Springer Journal of Information Retrieval Special Issue for ECIR 05
2. Vishwa Vinay, Ken Wood, Natasa Milic-Frayling, Ingemar Cox: **"Comparing Relevance Feedback Algorithms for Web Search"**: Poster at the World Wide Web (WWW) Conference 2005, Chiba, Japan
3. Vishwa Vinay, Ingemar Cox, Natasa Milic-Frayling, Ken Wood:**"Measuring the Complexity of a Collection of Documents"**: European Conference on Information Retrieval (ECIR) 2006, London, United Kingdom
4. Vishwa Vinay, Ingemar Cox, Natasa Milic-Frayling, Ken Wood:**"On Ranking the Effectiveness of Searches"**: 29th annual international ACM SIGIR conference on Research and development in information retrieval, Seattle, United States

Bibliography

[Aal92] I. K. Aalbersberg. Incremental relevance feedback. *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 11–22, 1992.

[ACR04] G. Amati, C. Carpineto, and G. Romano. Query difficulty, robustness and selective application of query expansion. In *Proceedings of the 25th European Conference on Information Retrieval*, 2004.

[AHK01] C. C. Aggarwal, A. Hinneburg, and D. A. Keim. On the Surprising Behavior of Distance Metrics in High Dimensional Space. *Lecture Notes in Computer Science*, 1973, 2001.

[Ani03] P. Anick. Using terminological feedback for web search refinement-a log -based study. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in Informaion Retrieval*, pages 88 – 95, 2003. Toronto, Canada.

[Ben69] R. S. Bennet. The Intrinsic Dimensionality of Signal Collections. *IEEE Transactions on Information Theory*, 15(5):517–525, 1969.

[BGMP00] O. Buyukkokten, H. Garcia-Molina, and A. Paepcke. Seeing the Whole in Parts: Text Summarization for Web Browsing on Handheld Devices. In *Proceedings of the Tenth International World Wide Web Conference*, 2000.

[BGMPW00] O. Buyukkokten, H. Garcia-Molina, A. Paepcke, and T. Winograd. Power Browser: Efficient Web Browsing for PDAs. In *Proceedings of the ACM Conference on Computers and Human Interaction*, 2000.

[BO97] D. Banks and R. Olszewski. Estimating Local Dimensionality. *Proceedings of the Statistical Computing Section of the American Statistical Association*, 1997.

[Bor03] P. Borlund. The iir evaluation model: a framework for evaluation of interactive information retrieval systems. *Information Research*, 8(3), 2003.

[BS98] J. Bruske and G. Sommer. Intrinsic Dimensionality Estimation With Optimally Topology Preserving Maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(5):572–575, 1998.

[BZJ99] M. W. Berry, Z. Drmac Z., and E. R. Jessup. Matrices, Vector Spaces, and Information Retrieval. *SIAM Review*, 41(2):335–362, 1999.

[CCR71] Y. K. Chang, C. Cirillo, and J. Razon. Evaluation of feedback retrieval using modified freezing, residual collection, and test and control groups. In G. Salton, editor, *The SMART Retrieval system - Experiments in Automatic Document Processing*. Prentice-Hall, 1971.

[CH79] W. Croft and D. Harper. Using probabilistic models of Information Retrieval without relevance information. *Journal of Documentation*, 35(4):285–295, 1979.

[CHR01] N. Craswell, D. Hawking, and S. E. Robertson. Effective site finding using link anchor information. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, 2001.

[CL76] T. F. Cox and T. Lewis. A conditional distance ratio method for analyzing spatial patterns. *Biometrika*, 63:483–491, 1976.

[CMK66] W. Cleverdon, C. J. Mills, and M. Keen. Aslib cranfield research project: Factors determining the performance of indexing systems. Technical report, Cranfield Institute of Technology, Cranfield, England, 1966.

[CMM+00] I. J. Cox, M. L. Miller, T. P. Minka, T. V. Papathomas, and P. N. Yianilos. The Bayesian Image Retrieval System, PicHunter: Theory, Implementation and Psychophysical Experiments. *IEEE Transactions on Image Processing*, 9(1):IEEE Transactions on Image Processing, 2000.

[CMZ03] Y. Chen, W-Y. Ma, and H-J. Zhang. Detecting Web Page Structure for Adaptive Viewing on Small Form Factor Devices. In *Proceedings of the Twelfth World Wide Web Conference*, 2003.

[CN01] E. Chavez and G. Navarro. Towards Measuring the Searching Complexity of General Metric Spaces. In *Proceedings of ENC'01*, 2001.

[CT91] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley-Interscience, 1991.

[CTZC02] S. Cronen-Townsend, Y. Zhou, and B. Croft. Predicting Query Performance. In *Proceedings of the 25th Annual International ACM SIGIR conference on Research and Development in Information Retrieval*, 2002.

[CvR96] I. Campbell and C. J. van Rijsbergen. The Ostensive model of developing information needs. In P. Ingwersen and N. O. Pors, editors, *Proceedings of CoLIS 2*, pages 251–268, 1996.

[CYTDP06] D. Carmel, E. Yom-Tov, A. Darlow, and D. Pelleg. What makes a query difficult? In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, 2006.

[Eft93] E. N. Efthimiadis. A user-centred evaluation of ranking algorithms for interactive query expansion. In E. Rasmussen R. Korfhage and P. Willett, editors, *Proceedings of the 16th Annual International Association of Computing Machinery, Specialist Interest Group in Information Retrieval*, pages 146–159, June 27 - July 1 1993.

[EHW87] A. El-Hamdouchi and P. Willett. Techniques for the measurement of clustering tendency in document retrieval systems. *Journal of Information Science*, 13(6):361–365, 1987.

[EKZ99] S. Epter, M. Krishnamoorthy, and M. Zaki. Clusterability Detection and Initial Seed Selection in Large Data Sets. Technical report, Rensselaer Polytechnic Institute, 1999.

[FO71] K. Fukunaga and D.R. Olsen. An Algorithm for finding intrinsic dimensionality of data. *IEEE Transactions on Computers*, 20(2):176–183, 1971.

[Har92] D. Harman. Relevance feedback revisited. *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 1–10, 1992.

[HB04] D. Harman and C. Buckley. The NRRC Reliable Information Access (RIA) Workshop. In *Proceedings of SIGIR 2004*, 2004.

[HO04] B. He and I. Ounis. Inferring Query Performance Using Pre-retrieval Predictors. In *Proceedings of the 11th Symposium on String Processing and Information Retrieval*, 2004.

[Hul96] D. A. Hull. Stemming algorithms: A case study for detailed evaluation. *Journal of the American Society of Information Science*, 47:70–84, 1996.

[HW67] H. A. Hall and N. H. Weiderman. The evaluation problems in relevance feedback systems. Report ISR-12 to the National Science Foundation, 1967. Department of Computer Science, Cornell University.

[Ide71] E. Ide. New experiments in relevance feedback. In G. Salton, editor, *The SMART retrieval system*, pages 337–354. Englewood Cliffs, NJ: Prentice-Hall, 1971.

[JD88] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall Advanced Reference Series, 1988.

[JM97] M. Jones and G. Marsden. From the Large Screen to the Small Screen: Retaining the Designer's Design for Effective user Interaction. In *IEEE*

Colloquium on Issues for Networked Interpersonal Communicators, volume 239(3), pages 1–4, 1997.

[JMMN+99] M. Jones, G. Marsden, N. Mohd-Nasir, K. Boone, and G. Buchanan. Improving Web Interaction on Small Displays. In *Proceedings of the 8th World Wide Web Conference*, 1999.

[Jon79] K. Sparck Jones. Search Term Relevance weighting given little relevance information. *Journal of Documentation*, 35(1):30–48, 1979.

[KB96] J. Koenemann and N. Belkin. A case for interaction: A study of interactive information retrieval behavior and effectiveness. In *Electronic Proeceedings of CHI '96*, 1996.

[Kwo05] K. L. Kwok. An Attempt to Identify Weakest and Strongest Queries. In *Workshop on Predicting Query Difficulty, Proceedings of SIGIR 2005*, 2005.

[KZ04] R. Kraft and J. Zien. Mining anchor text for query refinement. In *Proceedings of the 13th International World Wide Web Conference*, pages 666–674, 2004.

[LEM] LEMUR. The Lemur Toolkit for Language Modeling and Information Retrieval. http://www.lemurproject.org/.

[Lew95] D. Lewis. Active by accident: Relevance feedback in information retrieval. In AAAI Fall Symposium on Active Learning, 1995.

[LL99] M. Levene and G. Loizou. A probabilistic approach to navigation in Hypertext. *Information Sciences*, 114:165–186, 1999.

[Min00] T. P. Minka. Automatic Choice of Dimensionality for PCA. Technical Report 514, MIT Media Laboratory Perceptual Computing Section, 2000.

[Miz97] S. Mizzaro. Relevance: The Whole History. *Journal of the American Society of Information Science*, 48(9):810–832, 1997.

[ML02] T. Minka and J. Lafferty. Expectation-Propagation for the Generative Aspect Model. In *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence*, pages 352–359, 2002.

[MSN] MSN. MSN Search (http://search.msn.com).

[MvR97] M. Magennis and C. J. van Rijsbergen. The potential and actual effectiveness of interactive query expansion. *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 324–331, 1997.

[PBJD79] K. Pettis, T. Bailey, A. Jain, and R. Dubes. An intrinsic dimensionality estimator from near-neighbor information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1979.

[PD83] E. Panayirci and R. C. Dubes. A test for multidimensional clustering tendency. *Pattern Recognition*, 16(4), 1983.

[Pon98] J. Ponte. *A language modeling approach to information retrieval*. PhD thesis, University of Massachusetts at Amherst, 1998.

[Por80] M. F. Porter. An algorithm for suffix stripping. *Program*, 14:130–137, 1980.

[RL03] I. Ruthven and M. Lalmas. A survey on the use of relevance feedback for information access systems. *Knowledge Engineering Review*, 18(2):95–145, 2003.

[RMFSB03] K. Rodden, N. Milic-Frayling, R. Sommerer, and A. Blackwell. Effective Web Searching on Mobile Devices. In *Proceedings of the HCI Conference*, 2003.

[Rob97] S. E. Robertson. The probability ranking principle in IR. *Readings in information retrieval*, pages 281–286, 1997.

[Rob04] S. Robertson. Understanding inverse document frequency: on theoretical arguments for idf. *Journal of Documentation*, 60, 2004.

[Roc71] J. Rocchio. Relevance feedback informarian retrieval. In Gerard Salton, editor, *The Smart Retrieval System - Experiments in Automatic Document Processing*, pages 313–323. Prentice-Hall, Englewood Cliffs, N.J., 1971.

[RSJ76] S. E. Robertson and K. Sparck-Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27:129–146, 1976.

[Rut03] I. Ruthven. Re-examining the potential effectiveness of interactive query expansion. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, 2003.

[Ruv03] J-D. Ruvini. Adapting to the users internet search strategy. In *Proceedings of the Ninth International Conference on User Modeling*, pages 55–64, 2003.

[RWHB+95] S. E. Robertson, S. Walker, M. Hancock-Beaulieu, A. Gull, and M. Lau. Okapi at TREC-3. In D. Harman, editor, *Overview of the Third Text Retrieval Conference (TREC-3)*, 1995.

[Sal70] G. Salton. Evaluation problems in interactive Information Retrieval. *Information Storage and Retrieval*, 6(1):29–44, 1970.

[Sal75] G. Salton. *Theory of Indexing*. Society for Industrial and Applied Mathematics, 1975.

[SB88] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Journal of Information Processing and Management*, 24(5):513–523, 1988.

[SB90] G. Salton and C. Buckley. Improving retrieval performance by relevance feedback. In K. Sparck-Jones and E. Willet, editors, *Readings in Information Retrieval*. Morgan Kaufmann, New York, 1990.

[Shl05] J. Shlens. Tutorial on principal component analysis. http://www.snl.salk.edu/ shlens/pub/notes/pca.pdf, December 2005.

[SJ84] S. P. Smith and A. K. Jain. Testing for Uniformity in Multidimensional Data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6:73–81, 1984.

[SJWR00] K. Sparck-Jones, S. Walker, and S. E. Robertson. A probabilistic model of information retrieval: development and comparative experiments. *Information Processing and Management*, 36(6):809–840, 2000.

[SM97] G. Salton and M. J. McGill. The SMART and SIRE experimental retrieval systems. In K. Sparck Jones and E. Willet, editors, *Readings in Information Retrieval*, pages 381–399. Morgan Kaufmann, New York, 1997.

[SMS02] A. J. Sellen, R. Murphy, and K. L. Shaw. How knowledge workers use the web. In *Proceedings of the SIGCHI conference on Human factors in computing systems: Changing our world, changing ourselves*, pages 227–234, 2002.

[SvR83] A. F. Smeaton and C. J. van Rijsbergen. The Retrieval Effects of Query Expansion on a Feedback Document Retrieval System. *The Computer Journal*, 26(3):239–246, 1983.

[TB97] M. Tipping and C. Bishop. Probabilistic principal component analysis. Technical report, Neural Computing Research Group, Aston University, 1997.

[Tru76] G. V. Trunk. Statistical Estimation of the Intrinsic Dimensionality of a Noisy Signal Collection. *IEEE Transactions on Computers*, 25:165–171, 1976.

[TvR04] A. Tombros and C.J. van Rijsbergen. Query-sensitive similarity measures for Information Retrieval. *Knowledge and Information Systems*, 2004.

[VCMFW06] V. Vinay, I. Cox, N. Milic-Frayling, and K. Wood. On Ranking the Effectiveness of Searches. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and Development in Information Retrieval*, 2006.

[Voo03] E. M. Voorhees. Overview of the TREC 2004 Robust Retrieval Track. In *Proceedings of the 12th Text REtrieval Conference(TREC 2003)*, 2003.

[vR79] C. J. van Rijsbergen. *Information Retrieval*, volume Second Edition. Butterworths, London, 1979.

[Wil78] R. E. Williamson. Does relevance feedback improve document retrieval performance? *Proceedings of the 1st annual international ACM SIGIR conference on Information storage and retrieval*, pages 151–170, 1978.

[WJvRR04] R. W. White, J. M. Jose, C. J. van Rijsbergen, and I. Ruthven. A Simulated Study of Implicit Feedback Models. In *Proceedings of the 26th European Conference on IR Research (ECIR)*, pages 311–326, 2004.

[WR84] S. K. M. Wong and V. V. Raghavan. Vector space model of information retrieval: a reevaluation. In *Proceedings of the 7th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 167–185, 1984.

[WRJ02] R. W. White, I. G. Ruthven, and J. M. Jose. The use of implicit evidence for relevance feedback in web retrieval. 24th BCS-IRSG European Colloqium on IR Research, 2002.

[WWY92] Z. W. Wang, S. K. M. Wong, and Y. Y. Yao. An analysis of vector space models based on computational geometry. In *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 152–160, 1992.

[YTFCD05] E. Yom-Tov, S. Fine, D. Carmel, and A. Darlow. Learning to estimate query difficulty: including applications to missing content detection and distributed information retrieval. *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, 2005.

[ZH03] X. S. Zhou and T. S. Huang. Relevance feedback in image retrieval: A comprehensive review. *Multimedia Systems*, 8, 2003.

[Zip49] G. K. Zipf. *Human Behavior and the Principle of Least Effort*. Addison-Wesley, 1949.