Comparison of Blind Relevance Feedback Algorithms Using Controlled Queries

by
Christopher Jordan

Submitted in partial fulfillment of the
requirements for the degree of
Master of Computer Science

at

Dalhousie University
Halifax, Nova Scotia
March 2005

DALHOUSIE UNIVERSITY

FACULTY OF COMPUTER SCIENCE

The undersigned hereby certify that they have read and recommend to the Faculty of Graduate Studies for acceptance a thesis entitled "**Comparison of Blind Relevance Feedback Algorithms Using Controlled Queries**" by **Christopher Jordan** in partial fulfillment of the requirements for the degree of **Master of Computer Science**.

Dated: March 21, 2005

Supervisors:

Qigang Gao

Carolyn Watters

Reader:

Vlado Keselj

DALHOUSIE UNIVERSITY

Date: **March 21, 2005**

Author: **Christopher Jordan**

Title: **Comparison of Blind Relevance Feedback Algorithms Using Controlled Queries**

Department: **Computer Science**

Degree: **M.C.Sc.** Convocation: **May** Year: **2005**

Table of Contents

List of Tables

List of Figures

Abstract

One of the current issues with document retrieval is that it is not fully understood why some algorithms are better than others. Much of this confusion is due to evaluations typically being done on a set of user–defined queries. Such evaluations have little control over the amount of information in the query. Without this control it is difficult to know what are the causes of a given retrieval performance. Proposed here is a new evaluation method that addresses this lack of control; algorithms are tested on queries that are autonomously generated from predefined relevant sets of documents. Relative entropy is used to discover the most discriminating terms which are used to manufacture these queries. In this work, two blind relevance feedback (BRF) approaches are evaluated using these controlled queries. The results indicate that BRF does not improve retrieval performance for queries composed of only the most discriminating terms.

Acknowledgements

There are many people that have helped me in my journey through my Master's. I would like to first thank Dr. Qigang Gao and Dr. Carolyn Watters, my advisors, from Dalhousie University. Their attention to detail and passion for research have both inspired me and raised the caliber of my work. I also want to thank Major Mike Jackson and Mr. Mark Zawidzki from the Canadian Forces. They provided me with invaluable experiences in real world applications of knowledge management which has helped to shape and direct my work. Thanks is also warranted for the members of the Dalhousie Student Chapter of the ACM. They have been an excellent support group both providing helpful criticism and encouragement. Finally, I would like to thank my mother for the countless home cooked meals and the patience to be my sounding board for hours on end. Without her I surely would not have made it.

The research that was conducted for this thesis was funded by NSERC, grant number PGSM-302580-2004.

Chapter 1

Introduction

Presented in this thesis is a rigorous investigation of Blind Relevance Feedback (BRF), a technique that attempts to improve the performance of document retrieval systems by assuming that the top ranked documents are always relevant to the user. Traditionally retrieval systems are tested on queries authored by users. The fault with this practice is that there is very little control over the size of the user's search focus, their notion of relevancy, and the quality of queries they issue. This method of evaluation does indicate whether or not retrieval algorithms work and if the users are satisfied with them but it is difficult to understand why they work and why some are better than others in different situations. To address this lack of understanding, this work proposes a new approach for evaluating document retrieval algorithms that generates sets of queries of varying quality in a controlled manner. In this work, sets of these controlled queries are developed and tested on various retrieval algorithms incorporating BRF.

Document retrieval systems [5], popularly known as search engines, tackle an interesting problem domain where users formulate their information needs as queries. Search engines process these queries and retrieve documents accordingly in an attempt to satisfy them. Typical search engines process queries independently of users and return the same retrieved set to the ones that issue the same query. Here lies the weakness; this model assumes that users can adequately depict their information need in a query. Query formation, however, is a difficult task and very often high performance queries are long, complex, and unintuitive.

Users typically issue short queries consisting of 2 to 3 terms [43]. Such queries lack much information regarding the user need and consequently are ambiguous. This confusion is due to users having differing experiences, preferences, and education levels; factors such as these form what is known as the user context. As each user has their own unique context, they will have different ways of expressing their needs through

queries. This issue manifests itself as synonymy, many different terms having the same meaning, and polysemy, a term having many different meanings. Short queries simply lack the contextual information regarding the user to precisely define their information need. These queries are therefore ambiguous and cause poor retrieval performance.

As retrieval systems index more documents, the issue of user query formation becomes more serious. A lot of work has been done in order to help users overcome it though. Rocchio and Salton [35] published some of the first work in this area with the Rocchio relevance feedback algorithm. Relevance information is extracted directly from users in the form of relevance judgments or ratings; this type of feedback is known as explicit feedback. Since Rocchio's work there have been many different explicit feedback approaches developed. However it was discovered that, regardless of the approach, users tend to not give feedback. Apparently the reward of improved retrieval performance was not enough incentive for users. Work has been done on automatically extracting feedback from users. This relevance data is based on user behavior like the amount of time that one spends browsing documents; this type of feedback is known as implicit feedback [9]. However, it is interesting to note that recently Kelly *et al.* [22] have shown that the relationship between the time spent on a Web page and relevancy is not a direct one.

BRF is a straightforward approach to implicit feedback where after the user issues the initial query the top ranked documents in the retrieved set are assumed to be relevant and are automatically used as positive feedback in a second search. This technique was initially introduced during TREC–2 [15] and has since been the automatic feedback approach used in many works [8, 14, 17, 25, 27, 28, 49, 51] include the automatic query expansion feature in MySQL [29]. The simplicity of the approach, combined with its beneficial effects on retrieval performance, makes it very attractive. However, it makes two assumptions: the top ranked documents are relevant and they encompass the user's information need. Typically these assumptions do not hold as generally some of the top ranked documents will not be relevant and the top ranked documents usually do not fully describe the user's need. Non–relevant documents are referred to as noise. BRF has no mechanism for dealing with noise in the top ranked documents and consequently it is propagated in the feedback [27]. The root of this

issue is recognizing when the top ranked documents generally do represent the user information need and when they do not. If this can be discovered beforehand then it can be determined when BRF should be employed.

The documents in the retrieved set are dependent on the query. Thus, in order to determine when the top ranked documents are suitable for BRF, there has to be control over the quality of the queries during the evaluation. This thesis addresses this issue by proposing a new method for evaluating document retrieval algorithms by using queries that are generated based on a predefined target set of documents, a relevant set, where the amount of information each query contains is controlled. This new style of evaluation provides the beginnings for an investigation of why BRF works by identifying a set of queries that it should not be used for.

Hypothesis – Blind Relevance Feedback does not improve the retrieval performance of queries constructed from only the most discriminating terms, based on relative entropy, for a relevant set.

In order to create sets of queries in a controlled manner it has to be identified what terms most distinguish the relevant set from the corpus. This is accomplished by first creating language models [30, 44] of the relevant set and the corpus. Relative entropy [10] is then used to compare these two models to discover these discriminating terms. Further explanation of this controlled query generation process is reserved for chapter 3.

The proposed approach for evaluation can be used on any document retrieval algorithm. However, in this work, attention is given to BRF. While it has been shown that BRF does improve retrieval performance over algorithms not using feedback, very little is known as to why it works. A greater understanding of BRF has important consequences:

Knowing when to employ BRF – typically systems implementating BRF perform it for every query. If it can be detected when a user has issued an unsuitable query, BRF could be disabled preventing further decay in performance.

Discovering how to improve BRF – currently it is not well understood why BRF seems to improve performance in some instances and decreases it in others.

Greater comprehension of why it works can help guide the development of new algorithms employing BRF.

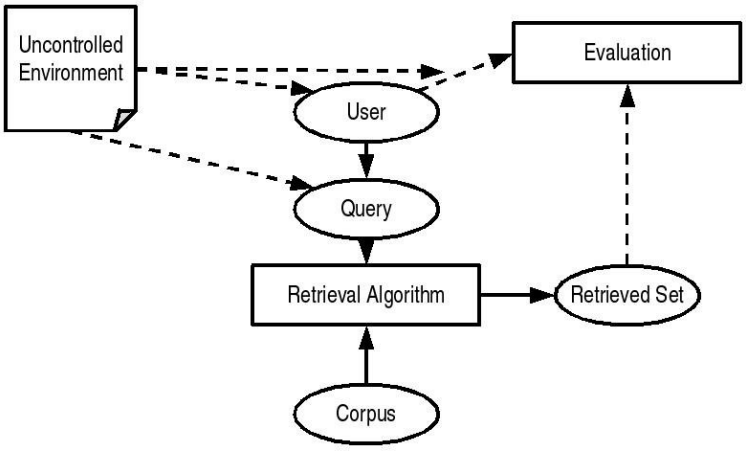

Figure 1.1: Conventional approach to evaluating document retrieval algorithms – In the conventional approach the user generates the query and is responsible for judging the relevancy of documents. In this approach there is very little control over how much information is contained in the query, the focus of the user's search, and the user's notion of relevance.

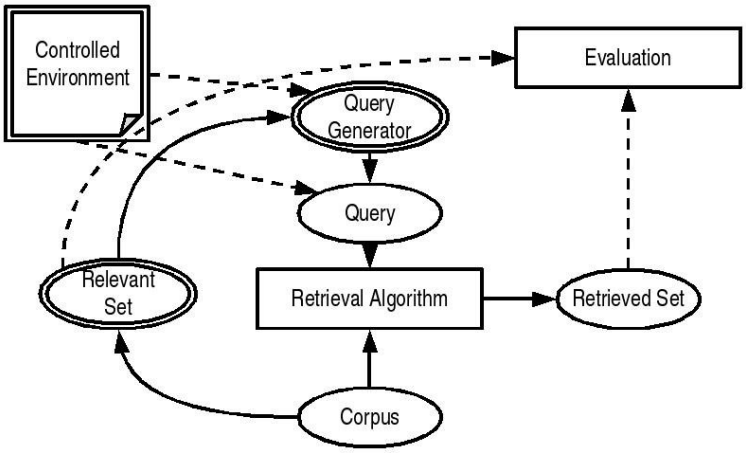

Figure 1.2: Proposed approach to evaluating document retrieval algorithms – In the approach proposed in this thesis, the user is removed from the query generation process. Queries are instead manufactured from predefined relevant sets of documents. This allows for direct control over how much information is contained in a particular query.

There are two contributions that this work makes. The first is a rigorous examination of the effects of employing BRF on retrieval algorithms. Previously this

technique has been implemented in retrieval systems and tested on user generated query sets in an *ad hoc* manner. Here, BRF is trialed on sets of synthetic queries that are increasing in quality created in a controlled manner. Though these trials are artificial in nature, this granularity is required in order to begin to truly understand why BRF works for some types of queries and fails for others and where it is best used.

The second contribution made here is a new technique for evaluating document retrieval algorithms. Sets of controlled test queries of varying quality are generated and used for the experimental trials. Previously, most system evaluations conducted involved users where they authored the queries. While human candidates may create more realistic queries, there is a lack of control and reproducability. The technique presented here provides a mechanism for controlling query quality thus providing a means not only to evaluate the performance of BRF but any retrieval algorithm in a regimented manner where query quality can be altered incrementally. The field of information retrieval currently is in need of this kind of evaluation [3].

Chapter 2

Background

The first two sections of this chapter present a review of classical document retrieval algorithms and metrics used to measure retrieval performance. These algorithms form the foundation for most work in document retrieval. The vector space model (VSM) in particular is one of the algorithms that is tested in the experimental trials presented in chapter 4. The following section discusses language modeling techniques that have been developed for information retrieval. Language modeling plays both a part in generating the queries used in the experimental trials and as a trialed retrieval algorithm. The fourth section is devoted to relevance feedback and blind relevance feedback (BRF) techniques as they are core components of this thesis. The final section of this chapter will be about relative entropy, a method for comparing probabilistic models. This thesis uses relative entropy in combination with language modeling to generate the query sets in a controlled manner. This process is discussed in Chapter 3.

2.1 Classical Document Retrieval Algorithms

There are three classical approaches to document retrieval [6]; they are Boolean, vector space, and probabilistic modeling. Boolean models retrieve documents based on whether or not they contain combinations of specified query terms. Queries for Boolean retrieval systems typically take the shape of Boolean expressions where the operands are terms. Documents which satisfy the Boolean expression query are consequently retrieved. The primary advantage of the Boolean model is that it is simple to implement and understand conceptually. The disadvantage of it is that there is no mechanism for ranking documents. Furthermore, forming complex Boolean queries is a demanding task for typical users.

VSM [40] creates a term document matrix (TD–matrix) in which each document is represented by a vector of term weights. Retrieval is conducted by treating the

query as a pseudo document and extracting the most similar document vectors in the TD–matrix. Probabilistic models, on the other hand, attempt to estimate the probability of any document being relevant to a query. The following two sub-sections discuss VSM and probabilistic modeling approaches further.

2.1.1 Vector Space Modeling

The VSM treats a corpus like a TD–matrix where each dimension represents a different term in the corpus. A document is represented by a particular term document vector. The values that a document has for each dimension in the vector are term weights calculated using term frequencies and inverse document frequencies. The comparison between two documents or a document and an *ad hoc* query is done using a similarity calculation, typically the cosine function which computes the angle between two vectors. Figure 2.1 gives a simple illustration of how the cosine similarity function works. The document and the query are considered to be two vectors in the vector space, in the case of figure 2.1 a 2 dimensional space. It is assumed that the smaller the angle between the document and query vectors the more similar they are to each other. The cosine of this angle gives a similarity value between 0 and 1 where 1 means that the document and query are exactly alike and 0 means they are complete opposites. It is important to note though that the vector space for a typical corpus has a large number of dimensions, usually in the range of thousands.

As mentioned previously, term weights for the dimensions of a vector are calculated by combining term and inverse document frequencies. The term frequency, tf, for a particular term is the number of times it occurs within the document in question as shown in equation 2.1. It is assumed that the more frequently a term occurs in a document, the more important it is. However, it is also recognized that some terms appear in more documents than others. The more documents that a term appears in, the less power it has to distinguish them. Stop words, words that occur in virtually every document but have little or no meaning like "the" and "at", are an example of these kinds of terms. The inverse document frequency, idf, is the log of the total number of documents in the corpus divided by the number of documents that contain the given term. The specifics of this calculation are shown in equation 2.2 Thus, the

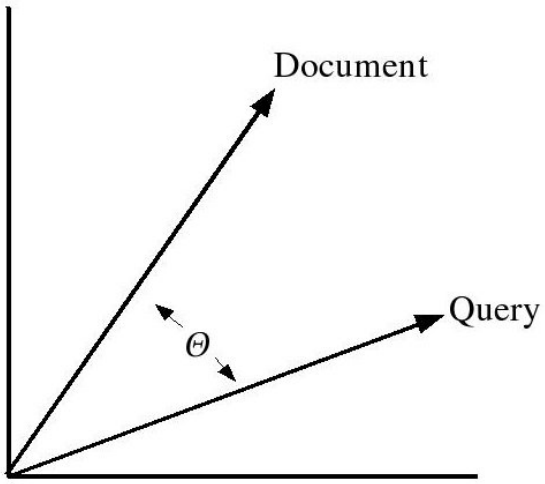

Figure 2.1: Cosine Similarity - the cosine similarity between a document and a query is the cosine of the angle between them.

fewer documents a term is in, the more important it is.

$$tf_{t,d_k} = \frac{freq_{t,d_k}}{\max(freq_{d_k})} \tag{2.1}$$

where t is a given term and d_k is a document in the corpus. $freq_{t,d_k}$ is the frequency of t in d_k and $freq_{d_k}$ is the set of term frequencies for d_k. This equation gives the normalized term frequency.

$$idf_t = \log \frac{N}{n_t} \tag{2.2}$$

where N is the total number of documents in the corpus and n_t is the number of documents that contain t. Term weight, w, given by equation 2.3 combines tf and idf by multiplying them together. In a sense, one can think of idf as a regulator for tf in the term weight calculation.

$$w_{t,d_k} = tf_{t,d_k} \times idf_t \tag{2.3}$$

The specifics of the cosine similarity calculation, sim_{VSM}, are below:

$$sim_{VSM}(d_k, q) = \frac{d_k \bullet q}{|d_k| \times |q|} = \frac{\sum_t w_{t,d_k} \times w_{t,q}}{\sqrt{\sum_t w_{t,d_k}^2 \times \sum_t w_{t,q}^2}} \tag{2.4}$$

VSM has been used as a baseline retrieval algorithm [40]. There have been several extensions to the VSM algorithm [4, 26, 52]. However it is interesting that after all this work there is no variant that clearly outperforms the original VSM for all circumstances.

2.1.2 Probabilistic Modeling

Robertson is responsible for some of the more popular work in probabilistic modeling. He, along with Sparck Jones [34], developed what is now known as the classical probabilistic model for information retrieval.

$$sim_{PM}(d_k, q) = \frac{P(R|d_k)}{P(\bar{R}|d_k)} \tag{2.5}$$

The above equation defines the similarity between a document d_k and a query q as the ratio between the posterior probabilities of it being relevant, R , or not, $\bar{R}$. The assumption is made that there exists a set of relevant documents for every query. This set will be referred to as the relevant set. The documents that have membership in this set should have the highest probability of being in it and the lowest probability for the contrary. For computational purposes, Bayes's rule is used to redefine similarity as:

$$sim_{PM}(d_k, q) = \frac{P(d_k|R) \times P(R)}{P(d_k|\bar{R}) \times P(\bar{R})} = \frac{P(d_k|R)}{P(d_k|\bar{R})} \tag{2.6}$$

The prior probabilities for relevancy, $P(R)$ and $P(\bar{R})$, are the same for every document thus can be ignored. In this model, documents can be viewed as a vector of terms. Assuming term independence, similarity can be computed based on the occurrence of query terms in the documents. Thus, the overall similarity of a document, d_k, to a query, q, can be calculated by the equation below.

$$sim_{PM}(d_k, q) = \frac{(\prod_{g(t,d_k)=1, t \epsilon q} P(t|R)) \times (\prod_{g(t,d_k)=0, t \epsilon q} P(\bar{t}|R))}{(\prod_{g(t,d_k)=1, t \epsilon q} P(t|\bar{R})) \times (\prod_{g(t,d_k)=0, t \epsilon q} P(\bar{t}|\bar{R}))} \tag{2.7}$$

The function $g(t, d_k)$ returns 1 if term t is present in document d_k and 0 if it is not. By taking the logarithm and replacing $P(\bar{t}|R)$ with $1 - P(t|R)$ and $P(\bar{t}|\bar{R})$ with $1 - P(t|\bar{R})$, the similarity can be computed as follows:

$$sim_{PM}(d_k, q) \sim \sum_{t \epsilon q} g(t, d_k) \times (\log \frac{P(t|R)}{1 - P(t|R)} + \log \frac{1 - P(t|\bar{R})}{P(t|\bar{R})}) \tag{2.8}$$

Hence we arrive at Robertson's recognized formula for the weight, w_t, of a term present in a document:

$$w_t = \log \frac{P(t|R)(1 - P(t|\bar{R}))}{P(t|\bar{R})(1 - P(t|R))} \tag{2.9}$$

This model also assumes that the relevant set can be adequately defined by query and document representations. However, in order to compute term weight some

estimation of the relevant set is required; such estimates can be achieved through relevance feedback, a technique explained in section 2.4. It is a common practice to pad these probabilistic estimations with a constant to avoid a probability of zero.

A drawback of this approach is that it does not consider the within document term frequency. This is addressed by Robertson *et al.* in their work on the best match (BM) series of algorithms, including BM25, which have been shown to have better retrieval performance over classical probabilistic modeling [31, 32]. The BM models are a direct descendent of this classical approach however further discussion of them is beyond the scope of this thesis.

2.2 Metrics

In the previous section, classical document retrieval algorithms were presented. It is from these algorithms that many current approaches find their roots. To compare document retrieval algorithms with each other there are a variety of standard metrics [6] that can be used. Metrics rate particular aspects of a retrieval algorithm such as how many documents in the retrieved set are relevant or how many out of the set of relevant documents were retrieved.

2.2.1 Precision and Recall

The most basic metrics in information retrieval are precision and recall. Precision is the ratio between the number of documents in the retrieved set that are relevant and the size of the retrieved set. Recall is the ratio between the number of relevant documents retrieved and the total number of relevant documents in the corpus. Essentially precision and recall measure accuracy and coverage of the retrieved set.

$$precision = \frac{|D_{relevant} \cap D_{retrieved}|}{|D_{retrieved}|} \tag{2.10}$$

$$recall = \frac{|D_{relevant} \cap D_{retrieved}|}{|D_{relevant}|} \tag{2.11}$$

where $D_{relevant}$ and $D_{retrieved}$ represent the set of relevant documents in the corpus and the retrieved set respectively. When examining the performance of a set of queries, there are two methods for calculating precision and recall; they are referred to as macro and micro. The macro approach computes a metric by taking its average

performance over the set of queries. The micro approach, instead of averaging, calculates a metric by using the total number of relevant documents retrieved and the total number of document retrieved for the entire set of queries.

One of the weaknesses with the precision and recall metrics is that they have to be measured at intervals. It is often necessary to interpolated precision and recall so that precision can be observed at different levels of recall. This quality makes it difficult for precision or recall to summarize the performance of a system. Another issue is that, with large corpora, it is difficult to determine the size of the relevant document set. Consequently, it may not be reasonable to measure recall leaving precision to be calculated over the top ranked documents. However, precision alone will not take into account the size of the relevant sets.

2.2.2 Mean Average Precision

The mean average precision (MAP) is a summarization metric. It is more specifically the mean of the average precision after relevant documents have been seen. Average precision, as shown in equation 2.12, is calculated by first summing the precision after each relevant document is seen in the retrieved set. Next, this value is divided by the total number of relevant documents that were retrieved.

$$average\ precision = \frac{\sum_{d_k \epsilon D_{relevant} \cap D_{retrieved}} precision\ at\ d_k}{|D_{relevant} \cap D_{retrieved}|} \tag{2.12}$$

MAP is computed by the equation below.

$$MAP = \frac{\sum_{q_j \epsilon Q} average\ precision_{q_j}}{|Q|} \tag{2.13}$$

where Q is the set of queries used to test the retrieval system. The advantage of this metric is that it offers a summary of how accurate the algorithm is over the test queries. The drawback is that it does not account for recall. So an algorithm can have a reasonably good MAP if it retrieves only a portion of the relevant documents so long as they are ranked high. Consequently this metric is often used with one that accounts for recall.

2.2.3 R–precision

R–Precision is a metric that complements MAP well as it depicts the overall recall of the top ranked documents. This is accomplished by measuring the precision of the

top R documents in the retrieved set where R is the number of relevant documents there are for the query. Of course, this means that the relevant set has to be well defined before this metric can be used properly.

$$R\text{–}Precision = \frac{|D_{relevant} \cap D_{top\ R}|}{|D_{top\ R}|} \tag{2.14}$$

$D_{top\ R}$ is the top R documents in the retrieved set. Obviously ideal performance, all the relevant documents are in the top rank, is achieved when R–Precision is high. If R-precision is low then it is useful to observe MAP to determine if relevant documents were simply ranked low or not retrieved at all.

2.3 Language Modeling

A relatively new approach to document retrieval is language modeling [30, 44]. It is a probabilistic approach to document retrieval though it is different from the classical approach as it does not attempt to segregate documents into relevant and non-relevant groups. Instead, language modeling derives models for every document. Documents are retrieved based on how likely their models would produce the query. A document model, more specifically, is the probability distribution of the vocabulary used by the corpus for a given document. This model can be approximated using the maximum likelihood estimate (MLE).

$$P_{ml}(t|d_k) = \frac{freq_{t,d_k}}{\sum freq_{d_k}} \tag{2.15}$$

where t is the term in question, d_k is a document in the corpus, $freq_{t,d_k}$ is the frequency with which t occurs in d_k, and $\sum freq_{d_k}$ is the sum of all the term frequencies in d_k. The problem with the MLE is that it assigns probabilities of zero to terms that do not occur in the document. This is a rather harsh assumption; just because a term does not occur in a document does not mean its model will never produce it. Thus some sort of smoothing has to be employed to get rid of these zero probabilities. Typically the probability of a term occurring in the corpus is used when the term is not present in a given document; this is referred to as a fallback probability. Hence the probability distribution for a document is:

$$P(t|d_k) = \begin{cases} P_s(t|d_k) & \text{where } t \text{ is in } d_k \\ \alpha P_{corpus}(t) & \text{otherwise} \end{cases} \tag{2.16}$$

where $P_s(t|d_k)$ is the smoothed probability of term t in document d_k, $P_{corpus}(t)$ is the probability of term t in the corpus, and α is a constant set heuristically that is typically less than 1 used to regulate how much of an effect $P_{corpus}(t)$ has on the model. $P_{corpus}(t)$ is typically approximated using the MLE. The reason $P_{corpus}(t)$ must be regulated is that the sum of all the probabilities in the model should be equal to one in order for it to be a probability mass function. In other words, smoothing should move probability mass from terms that have been seen in the document and allocate it to terms that have not. In [30] though, α was set to 1.

There are a variety of smoothing methods that can be employed for $P_s(t|d_k)$. In [30], a risk function is used to control how much the mean probability and maximum likelihood contribute. This function is based on how much the frequency for a given term differs from its average frequency over documents that contain it. In [44] a Good Turing approach was tried. In this technique, the probability for a given term is smoothed by a ratio of terms with similar frequencies in at least one document.

Zhai and Lafferty [50] examine three popular interpolation smoothing methods, Jelinek-Mercer, Bayesian smoothing using Dirichlet priors, and absolute discounting. The Jelinek-Mercer combines the MLE for a document directly with the corpus model while Dirichlet priors alters the term frequencies prior to the MLE calculation. Similarly, absolute discounting deducts a constant from all term frequencies before the MLE is computed. They found that smoothing generally improves retrieval performance. They also discovered that in general the Jelinek-Mercer and the Dirichlet priors smoothing methods tended to improve performance more than the absolute discounting method; Dirichlet priors performed the best on short queries while Jelinek-Mercer was the best on long ones. Table 2.1 summaries these three methods.

It should be noted that there is a small fault with language modeling. Terms that have stop word characteristics in a corpus may cause problems. If such a frequently used term is not present in a given document it may contribute more weight to the document model than if it was. Such instances are rare especially in heterogeneous corpora. However, it can occur in specific document sets and measures of some sort should be taken. As to what measures they might be, that is still an open question.

Smoothing Method	$P_s(t\|d_k)$	α	Comment
Jelinek-Mercer	$(1-\lambda)P_{ml}(t\|d_k) + \lambda P_{corpus}(t)$	λ	Performs well on long queries
Dirichlet priors	$\frac{c(t,d_k) + \lambda P_{corpus}(t)}{\|d_k\| + \lambda}$	$\frac{\lambda}{\|d_k\| + \lambda}$	Performs well on short queries
Absolute discounting	$\frac{\max(c(t,d_k) - \lambda, 0) + \lambda u(d_k) P_{corpus}(t)}{\|d_k\|}$	$\frac{\lambda u(d_k)}{\|d_k\|}$	Outperformed by Jelinek-Mercer and Dirichlet priors

Table 2.1: 3 popular interpolation smoothing methods - $u(d_k)$ is a function which returns the number of unique terms in document d_k.

An effective means of performing retrieval from these document models was developed by Song *et al.* [44] The query terms are considered to be a sequence of events. Documents are retrieved and ranked based on the probability that their models would generate the query as a random event as illustrated in equation 2.17

$$sim_{song}(d_k, q) = \prod_{t \in q} P(t|d_k) \tag{2.17}$$

Ponte *et al.* who first published the use of language modeling for document retrieval [30] developed a slightly different retrieval algorithm that does not incorporate query term frequencies. It was shown that Song's algorithm outperforms Ponte's and consequently will be one of the tested algorithms as described in chapter 4.

2.4 Relevance Feedback

The previous sections have discussed in some detail several document retrieval algorithms. The real issue that the algorithms all have to deal with is how to best interpret the user query in order to determine which documents they want. Document retrieval would be a relatively trivial task if users could simply list the identification numbers of the documents that they want to read. However, users generally do not know which documents they want. Users do know that they have an information need and that somehow they must convey that need to the retrieval system. Unfortunately, natural language queries are typically ambiguous. In this situation people are victims of their own ability to express themselves. There are many ways people can express

a particular information need in a query and consequently ambiguity can arise. This problem is further amplified by the tendency of users to issue short queries [43] even though it has been shown that using longer ones increases their satisfaction [7].

To cope with this problem of query formation Rocchio and Salton [35] introduced the concept of relevance feedback. Instead of issuing a query and then being left with the option of either accepting the retrieved set or modifying the query, the user can provide feedback on the retrieved documents. This feedback is used to modify the initial query. The steps of a typical feedback algorithm are as follows:

Step 1 The user issues a query and the corresponding retrieved set is returned

Step 2 The user provides judgments on the retrieved set

Step 3 These judgments are combined with the query by changing term weights and adding new terms. A new query, Q_{i+1}, is formed as result

Step 4 The new query is issued and the corresponding retrieved set is returned

Step 5 Repeat steps 2 - 4 until the user is satisfied with the retrieved set of documents

The Rocchio feedback algorithm requires users to make judgments on individual documents that are retrieved using the VSM. Feedback is incorporated into the search process using equation 2.18:

$$Q_{i+1} = \alpha Q_i + \beta \frac{\sum_{d \epsilon D_{pos}} d}{|D_{pos}|} - \gamma \frac{\sum_{d \epsilon D_{neg}} d}{|D_{neg}|} \tag{2.18}$$

Q_i and Q_{i+1} stand for the current and newly expanded queries respectively. D_{pos} and D_{neg} represent sets of documents judged to be relevant and not relevant by the user for the current query session. The α, β, and γ are constants that regulate how much impact the original query, positive feedback, and negative feedback have on the future query respectively. Ide [20] experimented with the number of relevance judgments incorporated into Q_{i+1} and differences between positive and negative feedback. Salton [39] conducted a survey of variants of the Rocchio algorithm and the constants to determine if there was a clear top performer. Both Ide and Salton found that there was no optimal amount of feedback or values for the constants however in all instances they found that feedback approaches improved overall retrieval performance. Salton also

offers an analysis of various feedback approaches for estimating $P(t|R)$ and $P(t|\bar{R})$ for probabilistic modeling.

Feedback does not necessarily have to be given on retrieved documents. In some recent work by Kruschwitz [23] users are asked to give judgments on concepts to narrow or broaden their queries. The HARD track at TREC [18] was started in 2003 in an effort to explore the incorporation of user contextual information into the search process. While approaches taken in 2003 failed to exploit stored user contextually information, there were a number of successful approaches developed for gathering feedback. Users were asked for feedback based on passages [33], sentences, noun phrases [47], and document clusters [1, 19, 42]. From the retrieval systems tested, it was found that judgments made on noun phrases were the most effective form of feedback. As well, systems that incorporated feedback on document clusters also experienced noticable improvements in retrieval performance.

The type of feedback which has been discussed thus far is classified as explicit feedback. This kind of feedback is given directly as user input, typically relevance judgments. Soliciting feedback from users poses a significant issue as users tend to minimize the amount of interaction that they have to do [41]. Without feedback these algorithms can not be used. Implicit feedback [9] attempts to avoid this issue by gathering relevance information based on user behavior. However, there is still much work left to determine a useful relation between user browsing behavior and the relevancy of the browsed material [22].

Allan found that in information filtering tasks only a small amount of feedback was required to notice substantial improvements [2]. In earlier work on contextual retrieval approaches [21], feedback is used to build user profiles to help enhance retrieval performance over time. The benefits of relevance feedback are obvious. However, it appears that the relation between feedback and improvements in retrieval performance is a non–linear one.

2.5 Blind Relevance Feedback

As described in the previous section, relevance feedback can significantly improve retrieval performance. The issue is getting that relevance feedback. Explicit feedback

encounters problems in soliciting it from users. Implicit feedback has issues in determining the relation between user behavior and document relevancy. Blind relevance feedback (BRF) avoids these issues all together by assuming that the top ranked documents are relevant and can be used as feedback. This technique is also known as pseudo relevance feedback, local feedback, and *ad hoc* feedback. BRF has received a lot of attention as it is a simple means of gathering relevance data that does not disrupt user searching behavior.

An issue with BRF is the assumption that the top ranked documents are relevant. Typically, there will be some non-relevant documents in the top rank. These documents are referred to as noise. Treating noise as relevant results in its propagation into the modified query. This situation is referred to as query drift [27].

Several approaches have been explored in an attempt to minimize query drift and improve BRF. Such experiments include varying the number of top ranked documents to use for feedback [28], varying the number of terms extracted from those documents [17], and using the locality of query terms within each document to determine which terms to extract [17, 49]. Locality is an attractive venue for exploration as it is hypothesized that it will help reduce the within document noise by ignoring sections that are not relevant to the user. The optimal number of documents and terms used for feedback generally depends on the query. This has lead to Sakai's work with optimization tables [36, 37, 38]. In this approach, a set of queries with known relevant document sets is used as training data. Sakai classifies these queries using various heuristics such as the number of search terms and the score of the highest ranking document. For each class of queries the optimal number of documents and terms to use for BRF is determined; these values are used to fill an optimization table. Queries issued by the user are then classified and the appropriate settings for BRF are looked up. While this approach has been shown to improve retrieval performance, it use of heuristics and training data may make it difficult to setup and maintain in a production environment.

Another approach that has been taken to help minimize query drift is re-ranking the initial set of retrieved documents [14, 27] before performing BRF. The idea behind this approach is that a document retrieval algorithm will generally retrieve some documents that are of interest to the user for a given query. In using a document

re–ranking algorithm to preprocess the retrieved set, it is hoped that more relevant documents can be discovered thus improving the feedback that is used for BRF.

An issue with the above BRF techniques is that all documents that are used for feedback are treated the same. However, based on the initial retrieved set of documents, not all such documents are necessarily equally relevant. Typical document retrieval algorithms associate a rank and a score with documents in the retrieved set. The score given by an algorithm represents how well the document meets the specifications of a particular query. Thus, the higher the score an algorithm assigns a document, the more relevant it should be. The argument can be made that documents with higher retrieval scores should contribute more to the feedback used in BRF.

2.5.1 Relevance Models

Lavrenko and Croft [11, 25] have used language modeling to develop a BRF system where the contribution of a feedback document depends on its retrieval score. This approach is referred to as relevance modeling. It uses the top ranked documents to estimate the relevancy of terms in Robertson's probability ranking principle:

$$\frac{P(d_k|R)}{P(d_k|\bar{R})} = \prod_{t \epsilon d_k} \frac{P(t|R)}{P(t|\bar{R})} \tag{2.19}$$

where $P(d_k|R)$ and $P(t|R)$ represent the probability of document d_k and term t given that they are relevant and $P(d_k|\bar{R})$ and $P(t|\bar{R})$ represent the probability of them given that they are not.

$$P(t|\bar{R}) \approx P_{corpus}(t) \tag{2.20}$$

$$P(t|R) \approx \frac{P(t, q_1, ..., q_n)}{P(q_1, ..., q_n)} \tag{2.21}$$

$$P(q_1, ..., q_n) = \sum_t P(t, q_1, ..., q_n) \tag{2.22}$$

The probability of t given that it is not relevant, $P(t|\bar{R})$, is estimated using the corpus model. The probability of it given that it is relevant, $P(t|R)$, is estimated in terms of $P(t, q_1, ..., q_n)$, the probability that the relevance model will generate the term t and the query terms $q_1, ..., q_n$ as a random event. Lavrenko *et al.* present two methods in their work for calculating this probability based on the top ranked

documents. The first, equation 2.23, assumes independence between the query terms and the terms in the vocabulary.

$$P(t, q_1, ..., q_n) = \sum_{d_k \in D_{TOP\_N}} P(d_k)P(t|d_k)\prod_{i=1}^{k} P(q_i|d_k) \tag{2.23}$$

where $P(t|d_k)$ and $P(q_i|d_k)$ are the probabilities of term t and query term q_i occurring given the term distribution derived from d_k, its language model. D_{TOP_N} is the set of top ranked documents in the retrieved set. The above computation is simply the sum of the probabilities for t being produced by the models for the top ranked documents regulated by the probability of these models producing the query terms; Lavrenko *et al.* refer to this approach as *Method 1*. The second approach, referred to as *Method 2* and, illustrated by equation 2.24, assumes a dependency between the query terms and terms in the vocabulary.

$$P(t, q_1, ..., q_n) = P(t)\prod_{i=1}^{n} P(q_i|t) \tag{2.24}$$

Lavrenko *et al.* estimates the probability of a query term, q_i, given a term, t, in the vocabulary with the following:

$$P(q_i|t) = \sum_{d_k \in D_{TOP\_N}} P(d_k|t)P(q_i|d_k) \tag{2.25}$$

Hence we have the following:

$$P(t, q_1, \ldots, q_n) = P(t)\prod_{i=1}^{n} \sum_{d_k \in D_{TOP\_N}} P(d_k|t)P(q_i|d_k) \tag{2.26}$$

These two approaches combine term distributions to form a new one. In Lavrenko's work, the top 50 ranked documents in the retrieved set are used for D_{TOP_N}. Not surprisingly, in the reported results the relative entropy, discussed in section 2.6, between the relevance models constructed in this manner and the models constructed from the true set of relevant documents is quite high. Lavrenko *et al.* was however still able to show that these models were able to improve retrieval and topic tracking performance over traditional language modeling on the retrieval and topic tracking TREC corpora.

2.6 Relative Entropy

The language modeling approach to information retrieval creates probabilistic models for every document in the corpus. Each document has a different probability distribution over the vocabulary. They are essential probability mass functions over a discrete and finite domain and so can be compared using relative entropy [10].

$$relative\,entropy(P(x), G(x)) = \sum_{x \in X} P(x) \log \frac{P(x)}{G(x)} \tag{2.27}$$

where $P(x)$ and $G(x)$ are two probability mass functions.

Relative entropy is not a true distance metric as it fails the triangle inequality and it is not symmetric. Suppose A, B, and C are three arbitary probability mass functions. Failing the triangle inequality means that the following inequality does not hold:

$$relative\,entropy(A, B) + relative\,entropy(B, C) >= relative\,entropy(A, C) \tag{2.28}$$

Not being symmetric means that the following equality does not hold:

$$relative\,entropy(A, B) = relative\,entropy(B, A) \tag{2.29}$$

Regardless, relative entropy can still be used to compare two probability distributions. Obviously relative entropy is very attractive for language modeling techniques in information retrieval as it provides a mechanism for comparing models to each other. The smaller the relative entropy between two models the more similar they are.

A direct application of relative entropy to document retrieval can be found in work done by Zhai and Lafferty [24]. Here models for both the query and the documents are estimated and then compared using relative entropy. The assumption is that documents that will satisfy the query will have language models similar to it. This approach is different from previous language modeling techniques proposed by Ponte [30] and Song [44] which the query was viewed as an event that could be produced by document models. The issue here is how to approximate the query models. Often times there is a very limited amount text data offered by the user. Zhai and Lafferty proposed the use of Markov chains to smooth query models assuming that it would add probability mass to terms that are semantically similar to the query terms. Zhai

and Lafferty continued this work by experimenting with approaches for incorporating relevance feedback [51] including one approach that generates a model that has the minimum relative entropy between all feedback document models. However, only BRF was explored in that work. Even so, their results indicate that retrieval performance for this approach is better than both standard language modeling and Rocchio algorithms.

An alternative use of relative entropy for BRF is automatic query expansion [8]. Instead of creating models of queries, a model is created from the top ranked documents. The terms that contribute the most to relative entropy between the model of the top ranked documents and the model of the corpus are then used for query expansion. In this work Cai *et al.* also explored other divergence scoring algorithms and found their performance to be comparable to relative entropy. The results from their trials show that this approach to query expansion may be more effective at improving precision than the Rocchio algorithm.

Relative entropy has also been used to measure the quality of a query. Cronen-Townsend *et al.* refer to the relative entropy between the model of the top ranked documents and the model of the corpus as the clarity score [12, 13]. In the trials that they conducted on several TREC corpora they discovered that the retrieval performance of a query is positively correlated to the clarity score. The advantage of this approach over other metrics is that it does not need to know the relevancy of documents in the retrieved set. However, in an analysis that Turpin and Hersh [45] conducted, it was found that there is no relation between clarity scoring and whether or not the user was successful in satisfying their information need during a search session. The clarity scores for the queries issued by the users in their study were relatively low and did not improve as users performed manual query modification.

2.7 Summary

In this chapter, a brief overview was given of some of the work that has been done in document retrieval. Classical algorithms such as VSM and probabilistic modeling form the basis of many current retrieval algorithms used today. As well, they are commonly used as baselines for comparison against newly developed approaches. Precision and recall are metrics that are sometimes used to conduct these comparisons

however they do not summarize the performance of an algorithm very well where as MAP and R–Precision do.

It was found that relevance feedback from the user can enhance retrieval performance. However, users typically do not divulge much feedback if any. Thus automatic methods for getting feedback were developed. One such popular method is BRF where the top ranked documents in a retrieved set are used as positive feedback.

Language modeling is a relatively new approach to document retrieval where probabilistic models are created for each document. Relevance modeling is a BRF technique for language modeling where a probabilistic model is constructed of the top ranked documents in the retrieved set. Relative entropy is an function for comparing two probability mass functions thus can be used to compare language models. BRF and query expansion techniques incorporating relative entropy have been explored.

For most of the work presented in this chapter, the comparative analyses that were conducted are based on user developed queries. The issue with such queries is that it is difficult to determine how good one is compared to another. This issue is due to the relevant set of documents for a user created query typically not being well defined. As well, it is difficult to get two user created queries that have the same relevant set. This lack of control over the queries used in analyses make it difficult to determine if the performance seen is due to the quality of the queries, size of the relevant sets, or the algorithm being examined. This work attempts to overcome this issue by proposing an approach for generating queries in a controlled manner. This process is presented in the next chapter.

Chapter 3

Methodology

The focus of this work is to identify conditions that cause blind relevance feedback (BRF) to fail to improve retrieval performance. Queries depict user information needs although how well queries articulate this need varies. To what extent one query varies from another is difficult to determine as the relevant sets are generally not well defined. There is a notion of query quality where high quality queries retrieve all the documents that satisfy the user's need without any noise. Lower quality queries will achieve this to a lesser degree resulting in either not all the relevant documents being retrieved or noise being mixed into the results or both. Users have been shown to author these low quality queries rather than good ones [43]. This is not surprising as creating a high quality query is more difficult than create low quality ones especially with large or unfamiliar corpora.

There are a variety of methods to help users overcome this issue. Most involve some level of user interaction beyond the initial query. The requirement for additional user interaction, especially feedback, is problematic as users tend to minimize their interactions and avoid relevance feedback interfaces [41]; BRF is a technique that avoids this problem. BRF assumes that the top ranked documents in a retrieved set are relevant to the user and uses them as feedback. Approaches that use BRF automatically modify user queries without requiring additional interaction.

Two related problems arise; first is the tendency that users have of issuing low quality queries and the second is the assumption that the top ranked documents are relevant. The low quality queries issued by the user typically return noise in the retrieved set thus questioning the validity of the assumption that BRF makes about the top ranked queries. Yet, BRF, discussed in chapter 2, still has been shown in previous work to improve retrieval performance thus demonstrating some robustness to noise. As query quality is directly related to the amount of noise which ultimately winds up in the retrieved set, this thesis proposes a novel techique for autonomously

generating queries in a controlled manner using relative entropy to vary the amount of information contained within them. The hypothesis is that BRF will fail to improve performance on queries that are composed of terms which most distinguish a relevant set from the corpus. The concept behind distinguishing terms can be traced back to Salton's notion of term discrimination defined in [40]. Relative entropy can be used to discover these discriminating terms which in turn can be used to manufacture queries of varying quality in a controlled manner.

3.1 The Queries

In order to test this hypothesis, a set of queries of varying quality is needed. While having users derive these sets does make them actual queries that a user would issue, such sets are difficult to obtain as not all users will come up with the same queries and their relevancy to the retrieved documents is uncertain. In addition, the notion of query quality for users is subjective. This is complicated further as the relevant document sets for such queries are often loosely defined on the corpora for which they are created as in the TREC HARD track for example.

Using automation to generate synthetic query sets has many attractive features. They are as follows:

1. Control – automation allows for measurable control over the quality of the queries which is not possible with user derived ones

2. Uniformity – relevant document sets are predefined and the query sets pertaining to them are generated. All query sets are created under the same conditions which is more difficult to do with users

3. Generation of queries on mass – automation allows for more queries to be create in a short amount of time

4. Reproducibility – as users are not creating the queries, the same queries sets are generated for the relevant sets each time

Whether automation or users are employed to create the query sets, they will be artificial in nature. Thus for the advantages listed above, automation is used instead of users to create the query sets.

3.1.1 Controlled Query Generation

Measuring the quality of a query has traditionally been done by processing it with a retrieval algorithm on a corpus and then using a performance metric, such as precision, to gage it. The issue with this method is that the target set of documents in the corpus has to be known for a query. This required knowledge makes it difficulty to rate a large number of queries. Clarity scoring [12, 13] is a metric that differs from the others in that it does not judge the quality of a query based on the relevancy of the retrieved documents. Clarity scoring works on the principle that the target set of documents is going to represent only a small fraction of the corpus. The retrieved document set and the corpus are represented probabilistically using language modeling and the clarity score is simply the relative entropy between the two.

In the controlled query generation approach that is presented here, the relative entropy between the relevant set and the corpus is used to derive a corresponding query set. This approach is broken into seven steps as illustrated in figure 3.1.

Step 1: Identify a relevant set

The first step in the process is to define a relevant set of documents. A detailed explanation of how these sets are defined for this work is in section 3.2.

Step 2: Define language models for the relevant set and the corpus

Let the relevant set of documents defined in step 1 be R. The language model for R is calculated by first concatenating all the documents in it together to form a single large document. The language model can then be determined easily using standard methods described in chapter 2. The Jelinek-Mercer smoothing method, discussed in section 2.3, was selected to be used in this work as it has been shown to be effective and relatively simple to implement.

$$R = d_1 \oplus d_2 \oplus d_3 \oplus \ldots \oplus d_n \tag{3.1}$$

$$P_{ml}(t|R) = \frac{freq_{t,R}}{\sum_i freq_{i,R}} \tag{3.2}$$

$$P_{jm}(t|R) = (1-\lambda)P_{ml}(t|R) - \lambda P_{ml}(t|corpus) \tag{3.3}$$

where $d_1, d_2, d_3, \ldots, d_n$ are documents which make up R as defined in step 1. $P_{ml}(t|R)$ and $P_{ml}(t|corpus)$ are the maximum likelihood estimates for R and the corpus. $P_{jm}(t|R)$ is the Jelinek-Mercer smoothing method.

The assumption that is made in creating this model is that all documents for a relevant set, R, are equally important and should contribute equally to the construction of the model. However, typically documents have a degree of relevancy to the user query.

Assumption: All documents in R are equally important. This supports their equal contribution to the construction of R's language model.

Step 3: Sort the terms in the vocabulary based on how much they contribute to relative entropy

Relative entropy can be used to measure the difference between two probabilistic models as described in section 2.6. As language models are probablistic models, the relative entropy between the model for R and the model for the corpus will reflect how different they are from each other.

$$relative\,entropy = \sum_{t \in V} P_{jm}(t|R) \log \frac{P_{jm}(t|R)}{P_{ml}(t|corpus)} \tag{3.4}$$

where t is a term in vocabulary V, the set of all terms in the corpus. The corpus model is estimated from all documents in the collection. It is assumed that the maximum likelihood estimate will adequately approximate it.

Assumption: The maximum likelihood estimate for the corpus represents its language model. This supports there being no need to smooth the corpus model.

The calculation of relative entropy is done iteratively over the vocabulary. Terms that contribute the most to relative entropy can be viewed as the terms that most distinguish the relevant set from the corpus. This allows for the

nature derivation of a term discrimination scoring function.

$$score(t) = P_{jm}(t|R) \log \frac{P_{jm}(t|R)}{P_{ml}(t|corpus)} \tag{3.5}$$

This approach is similar to Cai *et al.* [8] who used this scoring function and other divergence functions except they compared models of the top ranked documents to the corpus in order to find terms for query expansion.

Step 4: Identify the term that contributes the most to relative entropy and its average frequency in the relevant set. This term is the first generated query

The initial query is the term in the relevant set that contributes the most to relative entropy. The average frequency of this term in the relevant set is used as its query frequency.

Step 5: Identify the term that contributes the most to relative entropy and is not in a query

Step 6: Concatenate this term using its average frequency in the relevant set to the previously created query. Add this new query to the set of generated queries

Step 7: Repeat Steps 5-6 until there are no more terms

The above procedure creates queries exhaustively iterating through the vocabulary used by the corpus. For this work, this procedure is terminated when only terms that contribute less than 1% to relative entropy remain to be selected in step 5. The reason for this lower bound is that it was found in this work through experimentation that terms that contribute less than 1% to relative entropy have a negligible impact on the performance of the query.

3.2 The Relevant Sets

The previous section described a method for generating sets of queries from known relevant sets of documents. To support this method, a corpus with defined relevant

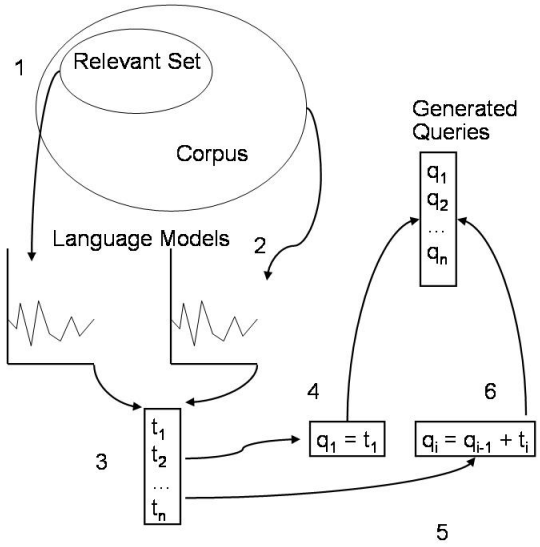

Figure 3.1: Controlled Query Generation.

sets has to be selected. The Reuters-21578 corpus [46] is one such document collection that meets this criteria. It is a small corpus that was developed by David Lewis and is composed of 21578 news articles that appeared on the Reuters newswire in 1987. Its documents have been rigorously categorized manually with a variety of thematic schemes. The relevant sets that are used in this work are defined using these categories. While there are other corpora that have queries sets constructed for them, the relevant document sets that pertain to them are often loosely defined. For example, the TREC HARD corpus has queries defined for it which have known relevant document sets. However, these relevant document sets were created by human evaluators judging only documents that were retrieved by predefined queries using various document retrieval systems. Thus the membership to these relevant sets has not been rigorously defined. The TREC HARD corpus is very large though and performing the same style of manual categorization that was done for Reuters-21578 would be both difficult and costly.

For the experimental trials conducted here, the relevant sets were defined using two factors in the schema, TOPICS and PLACES. Documents that have both a TOPICS and a PLACES label in common are considered to belong to the same relevant set. For example, "grain" is label in TOPICS and "canada" is a label in PLACES. The group of documents that have both the "grain" and "canada" labels are considered

to belong to the same relevant set. As documents in this dataset can have multiple TOPICS and PLACES labels, relevant sets are not exclusive.

The TOPICS scheme groups articles based on economic themes while the PLACES scheme groups them based on geographic themes. Articles can have multiple labels from TOPICS and PLACES although they are not required to have any. It should be noted that some articles in Reuters-21578 were not suitable for categorization by the TOPICS scheme and were marked with a BYPASS label. The LEWISSPLIT also gives documents a TRAIN, TEST, or NOT-USED label depending on whether they were used for training or testing or not used at all in experiments conducted by Lewis. Articles that have a BYPASS or NOT-USED label were not included as part of the retrievable corpus for any trials conducted in this thesis. As such, 19812 documents in Reuters-21578 were used for the experimental trials.

Chapter 4

Implementation

This chapter will address specific implementation details for the experimental trials. The first section discusses how the documents were preprocessed before the trials were run. The second section covers which retrieval algorithms were selected to be examined. The last two sections cover the performance metrics that were used and the actual design of the experiment.

4.1 Document Preprocessing

As described previously, individual documents are selected to be indexed from the Reuters-21578 corpus based on whether or not they are in a LEWISSPLIT. Although the documents are composed of different components, the only ones used in this work are the title and body which are concatenated and indexed. Stop words, *i.e.* terms that occur very frequently in the English language, are filtered out using a standard list [16]. This is a common procedure as stop words have very little information thus contribute very little to the overall content of the document. Inclusion of such terms tends to hurt retrieval performance of algorithms as they will typical be assign larger weights due to their within document frequencies. Stemming is also performed on the terms prior to indexing using Porter's algorithm [6]. Stemming is another common practice in document retrieval as it removes affixes from terms. Terms with a common base typically have very similar meanings thus this technique helps combat synonymy by reducing the effects affixes.

The unigram model is used for all the algorithms in this work. The unigram model simply requires that all terms that are indexed consist of a single word. This assumption however does not always hold as it ignores phrases, also known as polygrams. An examination of retrieval approaches using phrases is left for future work.

4.2 Retrieval Algorithms

There are five retrieval algorithms selected to be examined using the generated controlled queries. They are listed below. A description of them can be found in chapter 2.

- Vector Space Model
- Blind Relevance Feedback Rocchio using 5 documents
- Blind Relevance Feedback Rocchio using 10 documents
- Song's Language Modeling
- Relevance Modeling

Two Rocchio trials are conducted; one using the top 5 documents in the retrieved set as feedback and the other using the top 10. These two trials are BRF approaches to VSM. Relevance modeling is a BRF approach to language modeling. In the relevance modeling trial, the relevance models are built using the top 50 documents. This is consistent with Lavrenko's work [11, 25]. As well, Lavrenko's *Method 1*, explained in section 2.5.1, is used to estimate the relevance models. *Method 1* was choosen to be used because from Cronen-Townsend's work on clarity scoring it was reported to create better scores [12, 13]. The VSM based algorithms were selected for this experiment as they are a classical approach. The language modeling based algorithms were choosen because they are a relatively new approach to document retrieval.

The VSM algorithm used is the one described in chapter 2. Salton recommends the addition of constants to the term weight calculation [6]. However, it was observed in this work from experimentation that addition of these constants only improves the retrieval performance of short queries consisting of only a few terms. The constants actually have the oppose affect, harming performance, on queries containing many terms. As such, they are not incorporated into any of the VSM trials in order to be consistent.

For both the language and relevance modeling approaches, the Jelinek-Mercer method was used to perform smoothing. This method was chosen over others as it tends to allow for the greatest improve on long queries. Since the number of terms in

the queries created by the controlled query generation process described in chapter 3 increases rapidly, the Jelinek-Mercer method was the logical choice. The value for the λ constant is set to be .4 which is consistent with Cronen-Townsend *et al.* work on clarity scoring [12, 13].

4.3 Performance Metrics

Two performance metrics are used during the experimental trials, Mean Average Precision After Relevant Documents Seen (MAP) and R–Precision, described in sections 2.2.2 and 2.2.3. MAP is a commonly used performance metric and to an extent can attribute its popularity to TREC. One of the advantages of this metric is that it produces a single value for a retrieval algorithm's performance. MAP reports the average ranking of the relevant documents in the retrieved set. This metric however does not account for recall. Consequently, it is possible for a retrieved set to produce a high MAP value but have a low recall value.

To overcome the shortcoming of MAP, R–Precision is used synchronously. R–Precision complements MAP well as it indicates the degree of recall as the percentage of relevant documents in the top rank. For example, if R–Precision is low and MAP is high then only a few relevant documents have been retrieved though they are in the top rank. Another case is if R–Precision is low and MAP is low; this indicates that relevant documents have been retrieved but are not ranked high.

4.4 Experimental Design

There are a total of 1882 relevant sets that can be defined on the Reuters-21578 corpus using unique TOPICS and PLACES label pairs. Using the controlled approach to generating queries described in section 3.1.1, a total of 57660 queries were created over all the relevant sets. The purpose of this experiment is to test for query qualities where blind relevance feedback (BRF) fails to improve retrieval performance. Five different retrieval algorithms were explored as explained in section 4.2. They fall into two groups, VSM based ones and language modeling based ones. In each group, a BRF approach is examined.

For this work, all trials using BRF are limited to re–ranking the retrieved set

instead of performing a second retrieval. Document re–ranking was selected as the original retrieved set represents the set of documents that contains at least one of the original query terms; it is assumed in this work that documents that are relevant to the user must at least contain one of the original query terms. However, this assumption does not necessarily always hold as it is possible for a document to be relevant and not contain any query terms. Many terms suffer from synonymy or near synonymy and it is sometimes difficult for users to form a query where this assumption is true. While query expansion can help overcome this issue, terms also suffer from polysemy, having multiple meanings, and so expanded queries may retrieve non-relevant documents.

Assumption: Documents that are relevant for a particular query must contain at least one of the original query terms. This supports using BRF to only perform document re–ranking

Synonymy decays recall while polysemy decays precision. While query expansion partly addresses the issue of synonymy, it is hard to control the polysemous side effects. Further complications arises as heuristics have to be relied on to determine how many and which terms should be selected for query expansion. By restricting the BRF approaches to re–ranking the retrieved set, the emphasis of this work is on overcoming polysemy thus reducing ambiguity in the user query. Means for combating synonymy are left for future work.

Chapter 5

Results and Evaluations

For these experimental trials, a total of 1882 relevant sets are used; the largest containing 3143 documents. However, in the analysis presented in this section, the relevant sets "acq usa" and "earn usa" are ignored. This is due to the number of documents that they contain, 1787 and 3143 respectively, to be much more than the other relevant sets. The next largest relevant set is "grain usa" with a magnitude of 334. It was felt that by incorporating "acq usa" and "earn usa" into the statistical analyses that they would skew the results as they are outliers. Table 5.1 shows the distribution of relevant sets according to the number of documents they contain.

The randomized complete block design (RCBD) with a Tukey's honest significant difference (TukeyHSD) post hoc analysis[48] employing a 95% family wise confidence level was used on the collected results. The different algorithms in this experiment make up a 5 level treatment and individual queries are used as blocks. The queries can be used as blocks in this case as a block can be interpreted to consist of 5 queries that are exactly the same and are processed under exactly the same conditions varying only the algorithm used.

This analysis was conducted for both performance metrics, MAP and R–Precision. Groups of relevant sets based on size were examined. These groupings were used as it was observed that retrieval performance appears to depend on the magnitudes of the relevant sets. For this work, relevant set sizes from 1 to 10 are examined individually. As illustrated in table 5.1, there are fewer relevant sets that contain a large number of documents. In order to provide statistical significance during analysis, sets with sizes 11 to 15, 16 to 20, 21 to 50, and 51 to 400 are grouped together. To accelerate the computation of the RCBD for each group, without lost of generality, 100 queries selected randomly are used as the sample population. These queries are run on each algorithm.

The number of documents in a relevant set can be viewed as being representative

Size	Number of Relevant Sets
1	831
2	346
3	177
4	85
5	71
6	50
7	50
8	25
9	23
10	20
11 to 15	75
16 to 20	35
21 to 50	64
51 to 400	28

Table 5.1: Distribution of relevant sets according to size.

of different types of information seeking tasks that the user may have with document retrieval. The smaller magnitudes represent a more narrow search where the user is looking for specific documents. The large magnitudes are more illustrative of a broader search which is indicative of a user browsing documents.

The RCBD is run 4 times on each grouping of relevant sets. For each run, the random sample is drawn from a different subset of the generated queries. These subsets are:

- The entire set of generated queries
- The first third of generated queries
- The second third of generated queries
- The last third of generated queries

The sets of generated queries created by the process outlined in chapter 3 start with a single term query and each successive query has an additional term. Thus, the first third are the smallest queries and contain the least information. The last third are the largest queries and contain the most information. These different subsets of queries represent different levels of query quality.

The next two sections will present the outcome of the TukeyHSD analyses. The first table in each section lists the trials where the BRF approaches significantly outperform the baseline algorithms. The following tables in each section will list the converse, trials where the baseline algorithms significantly outperform the BRF approaches. To conserve space, the names for the algorithms are abbreviated as follows:

- Vector Space Model ⇒ VSM
- Blind Relevance Feedback Rocchio using 5 documents ⇒ roc5
- Blind Relevance Feedback Rocchio using 10 documents ⇒ roc10
- Song's Language Modeling ⇒ song
- Relevance Modeling ⇒ rele50

5.1 Mean Average Precision

The results of the RCBD are consistent over all relevant set groupings and query subsets; at least one of the 5 algorithms in each trial differed significantly from the rest with a probablity of a type I error less than 0.0001. A TukeyHSD post hoc analysis was conducted after each trial. These analyses reveal which algorithms differ from each other with statistical significants. Table 5.2 displays all the trials where the BRF approaches are significantly better than the baselines. Tables 5.3 and 5.4 show the trials where the baselines are significantly better than the BRF approaches. Appendix B contains a complete listing of the results from the TukeyHSD analyses on MAP.

roc10 > VSM		roc5 > VSM		rele50 > song	
Relevant set size	Query subset	Relevant set size	Query subset	Relevant set size	Query subset
None		21 to 50	1^{st} third	None	

Table 5.2: Trials where BRF is significantly better for MAP – This table shows the trials where the BRF approaches are significantly better than their baselines for MAP. This only occurs in one trial.

VSM > roc10		VSM > roc5		song > rele50	
Relevant set size	Query subset	Relevant set size	Query subset	Relevant set size	Query subset
1	All	1	All	2	All
1	1^{st} third	1	1^{st} third	2	1^{st} third
1	2^{nd} third	1	2^{nd} third	2	2^{nd} third
2	All	1	Last third	2	Last third
2	1^{st} third	2	All	3	All
2	2^{nd} third	2	1^{st} third	3	1^{st} third
2	Last third	2	2^{nd} third	3	2^{nd} third
3	All	2	Last third	3	Last third
3	1^{st} third	3	2^{nd} third	4	All
3	2^{nd} third	3	Last third	4	1^{st} third
3	Last third	4	2^{nd} third	4	2^{nd} third
4	All	4	Last third	4	Last third
4	2^{nd} third	5	2^{nd} third	5	All
4	Last third	5	Last third	5	1^{st} third
5	1^{st} third	6	2^{nd} third	5	2^{nd} third
5	2^{nd} third	6	Last third	5	Last third
5	Last third	7	Last third	6	All
6	1^{st} third	8	2^{nd} third	6	1^{st} third
6	2^{nd} third	8	Last third	6	2^{nd} third
6	Last third	9	Last third	6	Last third
7	Last third	11 to 15	Last third	7	All
8	1^{st} third	16 to 20	2^{nd} third	7	1^{st} third
8	2^{nd} third	16 to 20	Last third	7	2^{nd} third
8	Last third			7	Last third
9	2^{nd} third			8	All
9	Last third			8	1^{st} third
10	Last third			8	2^{nd} third
11 to 15	2^{nd} third			8	Last third
11 to 15	Last third			9	All third
16 to 20	2^{nd} third			9	2^{nd} third
16 to 20	Last third			9	Last third

Table 5.3: Trials where the baselines are significantly better for MAP Part 1 – This is one of two tables that shows the trials where the baselines are better than the BRF approaches.

song > rele50	
Relevant set size	Query subset
10	All third
10	1^{st} third
10	2^{nd} third
10	Last third
11 to 15	All third
11 to 15	1^{st} third
11 to 15	2^{nd} third
11 to 15	Last third
16 to 20	All third
16 to 20	1^{st} third
16 to 20	2^{nd} third
16 to 20	Last third
21 to 50	All third
21 to 50	1^{st} third
21 to 50	2^{nd} third
21 to 50	Last third
51 to 400	All third
51 to 400	1^{st} third
51 to 400	2^{nd} third
51 to 400	Last third

Table 5.4: Trials where the baselines are significantly better for MAP Part 2 – This is the second table that shows the trials where the baselines are significantly better than the BRF approaches for MAP. This table in particular shows some of the trials where language modeling is better than relevance modeling.

In these trials there is only one in which a BRF approach is significantly better than its baseline. On the contrary, there are many trials where the baselines are significantly better than the BRF approaches. In fact, there are 31 trials where VSM is significantly better than roc10 and 23 trails where it is significantly better than roc5. There are 51 trials where language modeling is significantly better than relevance modeling. These results provide support for the hypothesis that BRF does not improve retrieval performance on these types of queries.

Another important result that has come from the TukeyHSD analyses is that it appears that language modeling is significantly better than VSM for these queries. With the exception of two trials on relevant sets of size 1, language modeling was significantly better than VSM. This provides evidence that language modeling is better than VSM for these types of queries. Appendix B contains the results from all the TukeyHSD analyses using MAP as the dependent variable.

5.2 R–Precision

Like MAP, a RCBD was run on all relevant set groupings and query subsets. The results for every trial indicate that there is at least one algorithm with an R–Precision that is significantly different than the others with a probability of a type I error less than 0.0001. The following tables show the results of the TukeyHSD. Table 5.5 lists the trials where the BRF approaches are significantly better than the baselines. Tables 5.6 and 5.7 list the trials where the baselines are significantly better. Again the analyses point to BRF not improving performance. Appendix C contains a complete listing of the results from the TukeyHSD analyses on R–Precision.

roc10 > VSM		roc5 > VSM		rele50 > song	
Relevant set size	Query subset	Relevant set size	Query subset	Relevant set size	Query subset
None		None		None	

Table 5.5: Trials where BRF is significantly better for R–Precision – This table shows the trials where the BRF approaches are significantly better than their baselines for R–Precision. There are no trials where BRF is better.

VSM > roc10		VSM > roc5		song > rele50	
Relevant set size	Query subset	Relevant set size	Query subset	Relevant set size	Query subset
1	All	1	All	2	2^{nd} third
1	2^{nd} third	1	1^{st} third	2	Last third
1	Last third	1	2^{nd} third	3	All
2	All	1	Last third	3	1^{st} third
2	1^{st} third	2	All	3	2^{nd} third
2	2^{nd} third	2	1^{st} third	3	Last third
2	Last third	2	2^{nd} third	4	All
3	All	2	Last third	4	1^{st} third
3	2^{nd} third	3	2^{nd} third	4	2^{nd} third
3	Last third	3	Last third	4	Last third
4	2^{nd} third	4	2^{nd} third	5	All
4	Last third	11 to 15	2^{nd} third	5	1^{st} third
5	All	16 to 20	2^{nd} third	5	2^{nd} third
5	1^{st} third	16 to 20	Last third	5	Last third
5	2^{nd} third			6	All
5	Last third			6	1^{st} third
6	1^{st} third			6	2^{nd} third
6	2^{nd} third			6	Last third
6	Last third			7	All
7	2^{nd} third			7	1^{st} third
8	Last third			7	2^{nd} third
16 to 20	2^{nd} third			7	Last third
16 to 20	Last third			8	All
				8	1^{st} third
				8	2^{nd} third
				8	Last third

Table 5.6: Trials where the baselines are significantly better for R–Precision Part 1 – This is one of two tables that shows the trials where the baselines are better than the BRF approaches.

song > rele50	
Relevant set size	Query subset
9	All
9	1^{st} third
9	2^{nd} third
9	Last third
10	All
10	1^{st} third
10	2^{nd} third
10	Last third
11 to 15	All
11 to 15	1^{st} third
11 to 15	2^{nd} third
11 to 15	Last third
16 to 20	All
16 to 20	1^{st} third
16 to 20	2^{nd} third
16 to 20	Last third
21 to 50	All
21 to 50	1^{st} third
21 to 50	2^{nd} third
21 to 50	Last third
51 to 400	All
51 to 400	1^{st} third
51 to 400	2^{nd} third
51 to 400	Last third

Table 5.7: Trials where the baselines are significantly better for R–Precision Part 2 – This is the second table that shows the trials where the baselines are significantly better than the BRF approaches for R–Precision. This table in particular shows some of the trials where language modeling is better than relevance modeling.

The results of the trials measuring R–Precison are very similar to the results of the trials measuring MAP. There are no trials where a BRF approach is significantly better than its baseline. However, there are 23 trials where VSM is significantly better than roc10 and 14 trials where it is significantly better than roc5. Language modeling is significantly better than its BRF approach, relevance modeling, in 50 of the trials. As with MAP, these results provide support for the hypothesis that BRF does not improve retrieval performance on these types of queries.

The TukeyHSD also show that language modeling is significantly better than VSM on these queries. Similar to the analyses on MAP, language modeling is significantly better than VSM on every trial with the exception of two trials on relevant sets of size 1. Appendix C lists all the results from the TukeyHSD analyses using R–Precision as the dependent variable.

5.3 Summary

This chapter presented the results from the experimental trials conducted using queries generated by the controlled process discussed in chapter 3. This control allows for different relevant sets sizes and subsets of queries to be examined individually. Each combination of query subset and relevant set size represents a different search scenario or environment. There are a total of 64 different environments examined here and MAP and R–Precision are measured for each one on 5 different retrieval algorithms.

In these trials, typically the BRF approaches, roc10, roc5, and relevance modeling, do not outperform the baseline algorithms, VSM and language modeling. In fact, there are many trials where it appears the BRF is significantly worse than the baselines. These results support the hypothesis, stated in chapter 1, that BRF will not improve retrieval performance on queries composed of the most discriminating terms based on relative entropy. In other words, for most of the scenarios, represented by the trials, BRF failed to improve retrieval performance.

Another important result that was found from these trials is that Song's approach to language modeling was consistantly significantly better than Salton's VSM. In fact, there are trials where the difference in their retrieval performance is almost 100%. This evidence supports the use of language modeling over VSM for document

retrieval.

Chapter 6

Conclusions and Future Research

There are two contributions made by this thesis. The first is the introduction of a new approach to evaluating a document retrieval algorithm that, instead of using user generated queries, uses queries that are autonomously created in a controlled manner. One of the main issues with using user authored queries is that often times the true relevant set is not known prior to query generation. This makes it difficult to manually generate multiple queries for the same relevant set. Users have different contexts and as such will have different opinions as to what is relevant to queries they create. This leads to a lack of control over query quality during evaluation which in turn makes it hard to determine why a particular level of performance was achieved. Performance can be attributed to the quality of the query, size of the relevant set, users notion of relevance, and how distinguishable the relevant set is from the overall corpus. It is very difficult to determine and control these factors when involving users.

The automated approach presented in this thesis addresses this need for control. Contrary to a user evaluation where the relevant sets are defined *ad hoc* by the queries issued, a controlled query generation approach is proposed where relevant sets are known *a priori* and are used to define the queries. The queries are manufactured using the terms that contribute the most to the relative entropy between the language models for the relevant sets and the corpus. This allows for an experimental design where query quality and relevant set sizes can be varied. The drawback of this approach is that highly annotated corpora are required. For this level of annotation to be done to a corpus, a substantial amount of resources is required. Regardless, this approach gives the necessary control for simulating different types of queries and environments which algorithms can be exposed to. Conducting evaluations in this manner will help construct a better understanding of why certain algorithms are better than others in different circumstances.

This proposed method leads to the second contribution made by this thesis, providing evidence to support the hypothesis defined in chapter 1; blind relevance feedback (BRF) does not improve the retrieval performance of queries constructed from only the most discriminating terms, based on relative entropy, for a relevant set. In the experimental trials, BRF algorithms were evaluated to determine if they improve retrieval performance; BRF was limited to document re-ranking in this work. The results do indicate that BRF does not help on this type of query and in some cases hurts performance. This is contrary to many reported results as discussed in chapter 2. Query expansion techniques still have to be examined though their results may be similar to what was witnessed here. These examinations are left for future work due to the challenges with appropriately setting the heuristics involved as pointed out by Sakai [36, 37, 38].

Another important result is the significant difference in performance between language modeling and VSM. In the trials, language modeling consistantly significantly outperformed VSM. Further experimentation has to be done with different VSM based algorithms but this result provides a substantial case for using language modeling over VSM. An examination of probabilistic modeling approaches such as BM25 is left for future work.

This difference in performance between the VSM and language modeling approaches also provides justification for exploring the use of language modeling and relative entropy for contextual retrieval. More specifically, the approach used here for query generation is similar to how user profiles can be generated. It would be interesting to see if this approach can be extend to user profiling and if greater performance results are yielded.

A weaknesses with the proposed approach to query generation is that the method for constructing the queries represents only one type of query. Queries were formed by concatenating terms that most distinguish the relevant set based on relative entropy. This generates only a few queries out of the total possible. Further exploration using different methods for manufacturing queries has to be conducted. The current approach gives only one view of how the different retrieval algorithms perform on queries containing the most discriminating terms.

A second weakness in this approach to query generation is that it relies on relative

entropy to gage term discrimination. Relative entropy is an attractive function to use for calculating term discrimination as it takes into account both the probability of a terms occurring in the relevant set and in the corpus. However, relative entropy is not a true distance metric as it does not satisfy the triangle inequality [10]. Any extension to the query generation approach proposed here must consider this. It may be worth exploring alternatives to relative entropy as the term discrimination function.

A third weakness is the dependency of this approach on predefined relevant sets in the corpus. This requirement means that a corpus must be heavily categorized. Preprocessing a corpus in this manner is a very timely process as it requires human evaluators. To do this with large corpora such as the ones found in TREC will require a large effort. However, once a corpus is labeled, it never needs to be labeled again; it is a one time cost.

It is also important to consider the correlation between evaluations using users and evaluations using the controlled query generation approach proposed here. A positive relationship between the two styles would indicate that if an evaluation using controlled query generation identified one retrieval algorithm being better than another for a particular query environment, a similar result would be found in a user evaluation. Controlled query generation would not rule out the need for conducting user evaluations but instead minimize how many of them have to be conducted. This result is important considering that user evaluations tend to be expensive and time consuming. If there is a positive correlation, evaluations using controlled queries can be executed multiple times prior to a user evaluation in order to precisely fine tune retrieval algorithms or to determine if they are even worth trialing.

Overall, the method for creating controlled queries for evaluating and comparing document retrieval algorithms presented here makes a significant contribution addressing the lack of control in current evaluation techniques. This method is far from perfect though and there is still much work to be done regarding different methods on actually combining discriminating terms in query formation process. It is through different methods of generating these queries that different control environments can be setup which will ultimately bring to light different properties that retrieval algorithms may have. Evaluation using these controlled queries will lead to a better understand of document retrieval algorithms and consequently help guide future research and

development on them.

Bibliography

[1] N. AbdulJaleel, C. Corrada-Emmanuel, Q. Li, X. Liu, C. Wade, and J. Allan. Umass at TREC 2003: HARD and QA. In *TREC-12: Proceedings of the 12th annual Text REtrieval Conference*, 2003.

[2] James Allan. Incremental relevance feedback for information filtering. In *SIGIR '96: Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 270–278, 1996.

[3] James Allan, Jay Aslam, Nicholas Belkin, Chris Buckley, Jamie Callan, Bruce Croft, Sue Dumais, Norbert Fuhr, Donna Harman, David J. Harper, Djoerd Hiemstra, Thomas Hofmann, Eduard Hovy, Wessel Kraaij, John Lafferty, Victor Lavrenko, David Lewis, Liz Liddy, R. Manmatha, Andrew McCallum, Jay Ponte, John Prager, Dragomir Radev, Philip Resnik, Stephen Robertson, Roni Rosenfeld, Salim Roukos, Mark Sanderson, Rich Schwartz, Amit Singhal, Alan Smeaton, Howard Turtle, Ellen Voorhees, Ralph Weischedel, Jinxi Xu, and ChengXiang Zhai. Challenges in information retrieval and language modeling: report of a workshop held at the center for intelligent information retrieval, university of massachusetts amherst, september 2002. *SIGIR Forum*, 37(1):31–47, 2003.

[4] V. N. Anh and A. Moffat. Vector space ranking: Can we keep it simple? In *The Proceedings of the 7th Australian Document Computing Symposium*, pages 7–12, 2002.

[5] Arvind Arasu, Junghoo Cho, Hector Garcia-Molina, Andreas Paepcke, and Sriram Raghavan. Searching the web. *ACM Trans. Inter. Tech.*, 1(1):2–43, 2001.

[6] R. Baeza-Yates and G. Navarro. *Modern Information Retrieval*. Addison-Wesley, 1999.

[7] N. J. Belkin, D. Kelly, G. Kim, J.-Y. Kim, H.-J. Lee, G. Muresan, M.-C. Tang, X.-J. Yuan, and C. Cool. Query length in interactive information retrieval. In *SIGIR '03: Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, pages 205–212. ACM Press, 2003.

[8] D. Cai, C. J. van Rijsbergen, and J. M. Jose. Automatic query expansion based on divergence. In *CIKM '01: Proceedings of the Tenth International Conference on Information and Knowledge Management*, pages 419–426. ACM Press, 2001.

[9] Mark Claypool, Phong Le, Makoto Wased, and David Brown. Implicit interest indicators. In *Intelligent User Interfaces*, pages 33–40, 2001.

[10] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. John Wiley & sons, 1991.

[11] W. Bruce Croft, Stephen Cronen-Townsend, and Victor Larvrenko. Relevance feedback and personalization: A language modeling perspective. In *DELOS Workshop: Personalisation and Recommender Systems in Digital Libraries*, 2001.

[12] S. Cronen-Townsend and B. W. Croft. Quantifying query ambiguity. In *The Proceedings of HLT02*, 2002.

[13] Stephen Cronen-Townsend, Yun Zhou, and W. Bruce Croft. Predicting query performance. In *SIGIR '02: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Tampere, Finland*, pages 299–306, 2002.

[14] Carolyn J. Crouch, Donald B. Crouch, Qingyan Chen, and Steven J. Holtz. Improving the retrieval effectiveness of very short queries. *Information Processing and Management*, 38(1):1–36, 2002.

[15] D. Evans and R. Lefferts. Design and evaluation of the CLARIT–TREC–2 system. In *TREC2: Proceedings of the 2nd annual Text REtrieval Conference*, page 137, 1993.

[16] Glasgow IDOM - IR linguistic utilities: Stop word list. WWW `http://www.dcs.gla.ac.uk/idom/ir_resources/linguistic_utils/`, 2005.

[17] Z. Gu and M. Luo. Comparison of using passages and documents for blind relevance feedback in information retrieval. In *SIGIR '04: Proceedings of the 27st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Sheffield, UK*, pages 482–483. ACM Press, 2004.

[18] HARD project. WWW `http://ciir.cs.umass.edu/research/hard/`, 2005.

[19] D. He and D. Demner-Fushman. HARD experiment at maryland: from need negotiation to automated HARD process. In *TREC-12: Proceedings of the 12th annual Text REtrieval Conference*, 2003.

[20] E. Ide. New experiments in relevance feedback. In *The SMART Retrieval System: Experiments in Automatic Document Processing*, pages 337–354. 1971.

[21] Chris Jordan and Carolyn Watters. Extending the rocchio relevance feedback algorithm to provide contextual retrieval. In Jesús Favela, Ernestina Menasalvas Ruiz, and Edgar Chávez, editors, *Advances in Web Intelligence, Second International Atlantic Web Intelligence Conference, AWIC 2004, Cancun, Mexico, May 16-19, 2004. Proceedings*, volume 3034 of *Lecture Notes in Computer Science*, pages 135–144. Springer, 2004.

[22] Diane Kelly and Nicholas J. Belkin. Display time as implicit feedback: understanding task effects. In *SIGIR '04: Proceedings of the 27th Annual International Conference on Research and Development in Information Retrieval*, pages 377–384. ACM Press, 2004.

[23] Udo Kruschwitz. An adaptable search system for collections of partially structured documents. *IEEE Intelligent Systems*, 18(04):44–52, 2003.

[24] John Lafferty and Chengxiang Zhai. Document language models, query models, and risk minimization for information retrieval. In *SIGIR '01: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 111–119. ACM Press, 2001.

[25] Victor Lavrenko and W. Bruce Croft. Relevance based language models. In *SIGIR '01: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 120–127. ACM Press, 2001.

[26] Dik L. Lee, Huei Chuang, and Kent Seamons. Document ranking and the vector-space model. *IEEE Software*, 14(2):67–75, 1997.

[27] Mandar Mitra, Amit Singhal, and Chris Buckley. Improving automatic query expansion. In *SIGIR '98: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 206–214. ACM Press, 1998.

[28] J. Montgomery, L. Si, J. Callan, and D. A. Evans. Effect of vary number of documents in blind feedback: Analysis of the 2003 NRRC RIA Workshop "bf_numdocs" experiment suite. In *SIGIR '04: Proceedings of the 27st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Sheffield, UK*, pages 476–477. ACM Press, 2004.

[29] MySQL Reference Manual :: 12.6.2 full-text searches with query expansion. http://dev.mysql.com/doc/mysql/en/fulltext-query-expansion.html, 2005.

[30] Jay M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. In *SIGIR '98: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Melbourne, Australia*, pages 275–281. ACM Press, 1998.

[31] S. E. Robertson, S. Walker, and M. M. Hancock-Beaulieu. Large test collection experiments on an operational, interactive system: Okapi at TREC. In *TREC-2: Proceedings of the second annual Text REtrieval Conference*, pages 345–360. Pergamon Press, Inc., 1995.

[32] S. E. Robertson, S. Walker, K. Sparck Jones, M. M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC–3. In *TREC-3: Proceedings of the third annual Text REtrieval Conference*, 1994.

[33] S. E. Robertson, H. Zaragoza, and M. Taylor. Microsoft cambridge at TREC–12: HARD track. In *TREC-12: Proceedings of the 12th annual Text REtrieval Conference*, 2003.

[34] Stephen E. Robertson and Karen Sparck Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27(3):129–146, 1976.

[35] J. J. Rocchio. Relevance feedback in information retrieval. In *The SMART Retrieval System: Experiments in Automatic Document Processing*, pages 313–323. 1971.

[36] T. Sakai, S. E. Robertson, and S. Walker. Flexible pseudo-relevance feedback for NTCIR–2. In *The Proceedings of the Second NTCIR Workshop on Research in Chinese and Japanese Text Retrieval and Text Summarization*, 2001.

[37] Tetsuya Sakai, Masahiro Kajiura, and Kazuo Sumita. A first step towards flexible local feedback for ad hoc retrieval. In *IRAL '00: Proceedings of the fifth international workshop on on Information retrieval with Asian languages*, pages 95–102. ACM Press, 2000.

[38] Tetsuya Sakai and Stephen E. Robertson. Flexible pseudo-relevance feedback using optimization tables. In *SIGIR '01: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 396–397. ACM Press, 2001.

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

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

[41] Badrul M. Sarwar, Joseph A. Konstan, Al Borchers, Jonathan L. Herlocker, Bradley N. Miller, and John Riedl. Using filtering agents to improve prediction quality in the grouplens research collaborative filtering system. In *The Proceedings of the ACM Conference on Computer Supported Cooperative Work*, pages 345–354, 1998.

[42] J. G. Shanahan, J. Bennett, D. A. Evans, D. A. Hull, and J. Montgomery. Clairvoyance corporation experiments in the TREC 2003 high accuracy retrieval from documents (HARD) track. In *TREC-12: Proceedings of the 12th annual Text REtrieval Conference*, 2003.

[43] Craig Silverstein, Monika Henzinger, Hannes Marais, and Michael Moricz. Analysis of a very large altavista query log. Technical Report 1998-014, Digital SRC, 1998. http://gatekeeper.dec.com/pub/DEC/SRC/technical-notes/abstracts/src-tn-1998-014.html.

[44] Fei Song and W. Bruce Croft. A general language model for information retrieval. In *CIKM '99: Proceedings of the eighth international conference on Information and knowledge management*, pages 316–321. ACM Press, 1999.

[45] A. Turpin and W. Hersh. Do clarity scores for queries correlate with user performance? In *CRPIT '27: Proceedings of the Fifteenth Conference on Australasian Database*, pages 85–91. Australian Computer Society, Inc., 2004.

[46] UCI kdd archive: Reuters-21578 text catergorization collection. WWW `http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html`, 2005.

[47] O. Vechtomova, M. Karamuftuoglu, and E. Lam. Interactive search refinement techniques for HARD tasks. In *TREC-12: Proceedings of the 12th annual Text REtrieval Conference*, 2003.

[48] B.J. Winer, Donald R. Brown, and Kenneth M. Michels. *Statistical Principles in Experimental Design*. McGraw–Hill, third edition, 1991.

[49] S. Yu, D. Cai, J. Wen, and W. Ma. Improving pseudo-relevance feedback in web information retrieval using web page segmentation. Technical Report MSR-TR-2002-124, Microsoft Research, 2002.

[50] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *SIGIR '01: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 334–342, 2001.

[51] Chengxiang Zhai and John Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *CIKM '01: Proceedings of the Tenth International Conference on Information and Knowledge Management*, pages 403–410. ACM Press, 2001.

[52] Justin Zobel and Alistair Moffat. Exploring the similarity space. *SIGIR Forum*, 32(1):18–34, 1998.

Appendix A

Glossary

Boolean Model: An approach to document retrieval that retrieves documents that satisfy Boolean expression queries.

Blind Relevance Feedback: An approach for automatically gathering relevance feedback that assumes that the top ranked documents are relevant.

Corpus: A document collection.

Clarity Scoring: An approach for evaluating query quality by taking the relative entropy between the model for the retrieved set and the model for the corpus.

Explicit Feedback: Feedback that is given from the user as input.

Implicit Feedback: Feedback that is gathered from user behavior.

Language Model: A probability mass function representing the likelihoods of terms in a vocabulary being produced.

MLE: Maximum likelihood estimate.

Noise: Documents or information that is not relevant to the user's search.

Polysemy: The property of a term having multiple meanings.

Probabilistic Model: An approach to document retrieval that retrieves a given document based on the ratio between the probability that it is relevant and the probability that it is not.

RCBD: Randomized complete block design.

Relative Entropy: A divergence function that is used for comparing probability distributions.

Relevance Feedback: Feedback extracted from the user regarding the relevancy of the documents in the retrieved set.

Relevant Set: The set of documents that is relevant to the query.

Retrieved Set: The set of documents that is retrieved by a algorithm based on a query and corpus.

Synonymy: The property of many terms having the same meaning.

Target set of documents: The set of documents that is relevant to the query.

TukeyHSD: Tukey honest significant difference.

VSM: Vector space model.

Appendix B

MAP Results

This appendix contains the results from the Tukey's honest significant difference (TukeyHSD) post hoc analyses using MAP as the dependent variable. This analysis was performed on every trial as the results of the randomized complete block designs for each indicated that at least 1 of the 5 algorithms differed significantly from the others with a probablity of a type I error less than 0.0001. The TukeyHSD reveals which algorithms differ from each other with statistical significants. The results of these analyses are shown in the tables below; algorithms that do not differ significantly appear under the same line. The MAP for each algorithm in a trial appears directly below its label in the trial's corresponding table.

Query Set = All				
roc5	roc10	vsm	song	rele50
0.6575	0.6735	0.8288	0.9005	0.9176

Query Set = First Third				
roc5	roc10	vsm	rele50	song
0.6396	0.6629	0.7800	0.8793	0.8892

Query Set = Second Third				
roc5	roc10	rele50	vsm	song
0.8692	0.8894	0.9750	0.9800	0.9950

Query Set = Last Third				
roc5	roc10	rele50	vsm	song
0.9342	0.9574	0.9700	0.9950	1.0000

Table B.1: MAP results for relevant set size = 1 – In these trials roc5 and roc10 do not differ significantly nor do relevance modeling and language modeling. The only trial where VSM is not significantly better than both roc5 and roc10 is the one that samples from the last third.

Query Set = All				
roc10	roc5	vsm	rele50	song
0.4196	0.4546	0.5823	0.6563	0.7677

Query Set = First Third				
roc10	roc5	vsm	rele50	song
0.3497	0.3935	0.5119	0.6015	0.6981

Query Set = Second Third				
roc10	roc5	vsm	rele50	song
0.4994	0.5094	0.6621	0.6976	0.9097

Query Set = Last Third				
roc10	roc5	rele50	vsm	song
0.5716	0.6010	0.6662	0.7774	0.9107

Table B.2: MAP results for relevant set size = 2 – In these trials VSM is significantly better than roc5 and roc10. As well, language modeling is significantly better than relevance modeling and VSM.

Query Set = All				
roc10	roc5	vsm	rele50	song
0.3140	0.3358	0.3891	0.5021	0.6302

Query Set = First Third				
roc10	roc5	vsm	rele50	song
0.2616	0.3022	0.3567	0.4786	0.5885

Query Set = Second Third				
roc10	roc5	vsm	rele50	song
0.3447	0.3815	0.5009	0.5483	0.7415

Query Set = Last Third				
roc10	roc5	rele50	vsm	song
0.4353	0.4688	0.5765	0.6385	0.8271

Table B.3: MAP results for relevant set size = 3 – In these trials VSM is significantly better than roc10 however it is only significantly better than roc5 for trials sampling from the second and last third. Language modeling is significantly better than relevance modeling and VSM.

Query Set = All				
roc10	roc5	vsm	rele50	song
0.2990	0.3312	0.3667	0.4307	0.5441

Query Set = First Third				
roc10	roc5	vsm	rele50	song
0.2819	0.2972	0.3304	0.3893	0.4615

Query Set = Second Third				
roc10	roc5	vsm	rele50	song
0.3554	0.3899	0.4769	0.5301	0.7112

Query Set = Last Third				
roc10	roc5	rele50	vsm	song
0.3782	0.4480	0.4834	0.5386	0.7353

Table B.4: MAP results for relevant set size = 4 – Here, VSM is only significantly better than both roc5 and roc10 on trials sampling queries from the second and last third of the generated query sets. Language modeling is significantly superior to the other algorithms.

Query Set = All				
roc10	roc5	vsm	rele50	song
0.2924	0.3063	0.3424	0.4778	0.5966

Query Set = First Third				
roc10	roc5	vsm	rele50	song
0.3231	0.3518	0.3898	0.5004	0.6068

Query Set = Second Third				
roc10	roc5	rele50	vsm	song
0.4301	0.4830	0.5257	0.5796	0.7562

Query Set = Last Third				
roc10	rele50	roc5	vsm	song
0.4619	0.5037	0.5454	0.6501	0.7899

Table B.5: MAP results for relevant set size = 5 – VSM is significantly better than both roc5 and roc10 in the trials sampling queries from the second and last third of the generated query sets. Language modeling is significantly better than the other algorithms in all trials.

Query Set = All				
roc10	roc5	vsm	rele50	song
0.2355	0.2500	0.2839	0.4073	0.4887

Query Set = First Third				
roc10	roc5	vsm	rele50	song
0.2277	0.2548	0.2807	0.3724	0.4335

Query Set = Second Third				
roc10	roc5	vsm	rele50	song
0.2317	0.2674	0.3325	0.4774	0.6378

Query Set = Last Third				
roc10	roc5	vsm	rele50	song
0.2597	0.2951	0.3785	0.4723	0.7379

Table B.6: MAP results for relevant set size = 6 – In these trials roc10 and roc5 are not significantly different and VSM is only significantly better than both of them in the trials which sample from the second and last third of the generated query sets. Language modeling is significantly better than relevance modeling.

Query Set = All				
roc10	roc5	vsm	rele50	song
0.2262	0.2278	0.2512	0.4005	0.5422

Query Set = First Third				
roc10	roc5	vsm	rele50	song
0.1427	0.1465	0.1634	0.3079	0.4684

Query Set = Second Third				
roc10	roc5	vsm	rele50	song
0.1760	0.1810	0.2310	0.3966	0.6859

Query Set = Last Third				
roc10	roc5	vsm	rele50	song
0.1935	0.2088	0.2700	0.3732	0.7025

Table B.7: MAP results for relevant set size = 7 – VSM is only significantly better than roc10 and roc5 in the trial that samples queries from the last third of the generated query sets. Language modeling is significantly better than relevance modeling.

Query Set = All				
roc10	roc5	vsm	rele50	song
0.2486	0.2650	0.2877	0.4301	0.5625

Query Set = First Third				
roc10	roc5	vsm	rele50	song
0.2292	0.2521	0.2765	0.4207	0.5485

Query Set = Second Third				
roc10	roc5	vsm	rele50	song
0.2615	0.2864	0.3568	0.4284	0.7106

Query Set = Last Third				
roc5	roc10	vsm	rele50	song
0.3222	0.3297	0.4323	0.4424	0.7559

Table B.8: MAP results for relevant set size = 8 – Roc10 and roc5 are not significantly different. VSM is significantly better than both roc10 and roc5 in the trials that sample from the second and last third of the generated query sets. Language modeling is significantly better than both relevance modeling and VSM.

Query Set = All				
roc10	roc5	vsm	rele50	song
0.2002	0.2106	0.2280	0.3649	0.4213

Query Set = First Third				
roc10	roc5	vsm	rele50	song
0.1641	0.1693	0.1852	0.3250	0.3750

Query Set = Second Third				
roc10	roc5	vsm	rele50	song
0.2106	0.2273	0.2638	0.4154	0.5733

Query Set = Last Third				
roc10	roc5	vsm	rele50	song
0.2309	0.2369	0.3043	0.4082	0.6279

Table B.9: MAP results for relevant set size = 9 – In these trials roc10 and roc5 are not significantly different. VSM is not significantly different than roc5 in all trials except the one that samples from the last third of the generated query sets. Language modeling significantly outperforms relevance modeling in all trials except the one that samples from the first third of the generated query sets.

Query Set = All				
roc10	roc5	vsm	rele50	song
0.1829	0.2043	0.2097	0.3243	0.4149

Query Set = First Third				
roc10	roc5	vsm	rele50	song
0.1835	0.1957	0.2022	0.2844	0.3961

Query Set = Second Third				
roc10	roc5	vsm	rele50	song
0.2075	0.2238	0.2490	0.3211	0.5160

Query Set = Last Third				
roc10	roc5	vsm	rele50	song
0.2407	0.2473	0.2868	0.3474	0.5792

Table B.10: MAP results for relevant set size = 10 – In these trials roc5 is not significantly different from VSM. Language modeling is significantly better than relevance modeling and VSM.

Query Set = All				
roc5	roc10	vsm	rele50	song
0.3321	0.3237	0.3331	0.4165	0.5117

Query Set = First Third				
roc10	roc5	vsm	rele50	song
0.2550	0.2694	0.2803	0.3409	0.4704

Query Set = Second Third				
roc10	roc5	vsm	rele50	song
0.2380	0.2480	0.2873	0.3934	0.6321

Query Set = Last Third				
roc10	roc5	vsm	rele50	song
0.2128	0.2185	0.2700	0.3900	0.6813

Table B.11: MAP results for relevant set size = 11 to 15 – Here, VSM is only significantly better than both roc5 and roc10 in the trial that samples from the last third of the generated query sets. Language modeling is significantly better than the other algorithms.

Query Set = All					Query Set = First Third				
roc5	vsm	roc10	rele50	song	roc5	roc10	vsm	rele50	song
0.2179	0.2189	0.2204	0.2910	0.4071	0.1925	0.1940	0.1996	0.2815	0.3970
Query Set = Second Third					Query Set = Last Third				
roc10	roc5	vsm	rele50	song	roc10	roc5	vsm	rele50	song
0.1646	0.1687	0.2095	0.3481	0.5341	0.1978	0.2021	0.2500	0.3815	0.5655

Table B.12: MAP results for relevant set size = 16 to 20 – VSM is only significantly better than both roc10 and roc5 in the trials that sample from the second and last third of the generated query sets. Language modeling is significantly better than the other algorithms.

Query Set = All					Query Set = First Third				
vsm	roc10	roc5	rele50	song	vsm	roc10	roc5	rele50	song
0.2365	0.2444	0.2456	0.3194	0.4071	0.2836	0.2933	0.2958	0.3314	0.4206
Query Set = Second Third					Query Set = Last Third				
roc5	roc10	vsm	rele50	song	roc5	roc10	vsm	rele50	song
0.2769	0.2788	0.2830	0.3298	0.5304	0.2572	0.2596	0.2782	0.3271	0.5426

Table B.13: MAP results for relevant set size = 21 to 50 – VSM is not significantly different from both roc10 and roc5 in all trials except in the one that samples from the first third of the generated query sets; here roc5 is significantly better. Language modeling is significantly better than the other algorithms.

Query Set = All				
vsm	roc5	roc10	rele50	song
0.2791	0.2856	0.2879	0.3032	0.4363

Query Set = First Third				
vsm	rele50	roc10	roc5	song
0.3058	0.3069	0.3184	0.3185	0.4366

Query Set = Second Third				
roc5	roc10	vsm	rele50	song
0.2520	0.2566	0.2650	0.2730	0.4955

Query Set = Last Third				
roc5	roc10	vsm	rele50	song
0.2522	0.2546	0.2695	0.3105	0.5236

Table B.14: MAP results for relevant set size = 51 to 400 – Here, VSM is not significantly different from roc10 or roc5. Language modeling is significantly better than relevance modeling.

Appendix C

R–Precision Results

This appendix contains the results from the Tukey's honest significant difference (TukeyHSD) post hoc analyses using R–Precision as the dependent variable. The TukeyHSD was conducted on each trial as the randomized complete block designs for each indicated that at least 1 of the 5 tested algorithms are significantly different with a probability of a type I error less than 0.0001. The following tables show the results of the TukeyHSD on each trial. Algorithms that are not significantly different from each other appear under the same line. The average R–Precision for an algorithm in a trial appears below its label in the trial's corresponding table.

Query Set = All					Query Set = First Third				
roc5	roc10	vsm	rele50	song	roc5	roc10	vsm	song	rele50
0.42	0.48	0.65	0.89	0.89	0.44	0.50	0.62	0.79	0.81
Query Set = Second Third					Query Set = Last Third				
roc5	roc10	vsm	rele50	song	roc5	roc10	vsm	rele50	song
0.73	0.74	0.97	0.97	0.98	0.90	0.91	1.00	1.00	1.00

Table C.1: R–Precision results for relevant set size = 1 – In these trials roc5 and roc10 are not significantly different. VSM is significantly better than them except in the trial that samples queries from the first third. Language modeling and relevance modeling are not significantly different.

Query Set = All				
roc10	roc5	vsm	rele50	song
0.370	0.370	0.515	0.605	0.655

Query Set = First Third				
roc10	roc5	vsm	rele50	song
0.330	0.380	0.530	0.635	0.720

Query Set = Second Third				
roc10	roc5	vsm	rele50	song
0.400	0.420	0.580	0.635	0.805

Query Set = Last Third				
roc10	roc5	vsm	rele50	song
0.555	0.580	0.770	0.670	0.915

Table C.2: R–Precision results for relevant set size = 2 – VSM is significantly better than roc10 and roc5. Language modeling is only significantly better than relevance modeling for trials sampling queries from the second and last third.

Query Set = All				
roc10	roc5	vsm	rele50	song
0.2200	0.2633	0.3200	0.4500	0.5333

Query Set = First Third				
roc10	roc5	vsm	rele50	song
0.2667	0.3100	0.3333	0.4400	0.5333

Query Set = Second Third				
roc10	roc5	vsm	rele50	song
0.3100	0.4033	0.4967	0.5167	0.6800

Query Set = Last Third				
roc10	roc5	rele50	vsm	song
0.4000	0.4633	0.5133	0.5900	0.7500

Table C.3: R–Precision results for relevant set size = 3 – VSM is only significantly better than both roc5 and roc10 for trials sampling queries from the second and last third. Language modeling is significantly better than the other algorithms over all trials.

Query Set = All				
roc10	roc5	vsm	rele50	song
0.2825	0.2975	0.3250	0.4150	0.5000

Query Set = First Third				
roc10	roc5	vsm	rele50	song
0.2175	0.2400	0.2650	0.3350	0.4000

Query Set = Second Third				
roc10	roc5	vsm	rele50	song
0.3200	0.3700	0.4450	0.4825	0.6350

Query Set = Last Third				
roc10	roc5	rele50	vsm	song
0.3450	0.4350	0.4375	0.4775	0.6575

Table C.4: R–Precision results for relevant set size = 4 – VSM is only significantly better than both roc5 and roc10 in the trial that samples from the second third of the generated query sets. Language model is significantly better than the other algorithms.

Query Set = All				
roc10	roc5	vsm	rele50	song
0.254	0.310	0.316	0.396	0.488

Query Set = First Third				
roc10	roc5	vsm	rele50	song
0.292	0.366	0.374	0.478	0.566

Query Set = Second Third				
roc10	roc5	vsm	rele50	song
0.336	0.448	0.488	0.504	0.708

Query Set = Last Third				
roc10	rele50	roc5	vsm	song
0.434	0.496	0.546	0.574	0.754

Table C.5: R–Precision results for relevant set size = 5 – Roc5 is not significantly different from VSM in any trial. As well, language modeling is significantly better than the other algorithms.

Query Set = All				
roc10	roc5	vsm	rele50	song
0.2167	0.2467	0.2533	0.3717	0.4583

Query Set = First Third				
roc10	roc5	vsm	rele50	song
0.1833	0.2433	0.2450	0.3500	0.4067

Query Set = Second Third				
roc10	roc5	vsm	rele50	song
0.2083	0.2833	0.3017	0.4317	0.5717

Query Set = Last Third				
roc10	roc5	vsm	rele50	song
0.2283	0.3150	0.3367	0.4617	0.6817

Table C.6: R–Precision results for relevant set size = 6 – Roc5 and VSM are not significantly different in these trials. Language modeling is significantly better than the other algorithms.

Query Set = All				
roc10	roc5	vsm	rele50	song
0.1871	0.2143	0.2257	0.3586	0.4986

Query Set = First Third				
roc10	roc5	vsm	rele50	song
0.1286	0.1471	0.1600	0.2871	0.4286

Query Set = Second Third				
roc10	roc5	vsm	rele50	song
0.1329	0.1671	0.2114	0.3400	0.6071

Query Set = Last Third				
roc10	roc5	vsm	rele50	song
0.1743	0.2257	0.2343	0.3686	0.6643

Table C.7: R–Precision results for relevant set size = 7 – Roc5 and VSM are not significantly different. As well, roc10 and VSM are only significantly different on the trial that samples queries from the second third. Language modeling is significantly better than the other algorithms.

Query Set = All					Query Set = First Third				
roc10	roc5	vsm	rele50	song	roc10	roc5	vsm	rele50	song
0.2538	0.2650	0.2775	0.3875	0.4763	0.2288	0.2387	0.2512	0.3825	0.4712
Query Set = Second Third					Query Set = Last Third				
roc10	roc5	vsm	rele50	song	roc10	roc5	vsm	rele50	song
0.3150	0.3200	0.3588	0.4412	0.6512	0.3450	0.3513	0.3987	0.4425	0.6788

Table C.8: R–Precision results for relevant set size = 8 – Roc5 and VSM are not significantly different. As well, roc10 and VSM are not significantly different except on the trial that samples from the last third of the generated query sets. Language modeling is significantly better than the other algorithms.

Query Set = All					Query Set = First Third				
roc10	roc5	vsm	rele50	song	roc10	roc5	vsm	rele50	song
0.1767	0.1944	0.2000	0.3244	0.3844	0.1411	0.1578	0.1744	0.2844	0.3433
Query Set = Second Third					Query Set = Last Third				
roc10	roc5	vsm	rele50	song	roc10	roc5	vsm	rele50	song
0.2344	0.2433	0.2811	0.3867	0.5356	0.2378	0.2444	0.2889	0.3700	0.5478

Table C.9: R–Precision results for relevant set size = 9 – Roc10, roc5, and VSM are not significantly different on these trials. Language modeling is significantly better than the other algorithms.

Query Set = All				
roc10	roc5	vsm	rele50	song
0.207	0.214	0.224	0.339	0.421

Query Set = First Third				
roc10	roc5	vsm	rele50	song
0.189	0.189	0.196	0.274	0.357

Query Set = Second Third				
roc5	roc10	vsm	rele50	song
0.242	0.255	0.263	0.351	0.495

Query Set = Last Third				
roc5	roc10	vsm	rele50	song
0.280	0.283	0.303	0.359	0.555

Table C.10: R–Precision results for relevant set size = 10 – Here, VSM, roc10, and roc5 do not differ significantly. Language model is significantly better than relevance modeling.

Query Set = All				
roc5	roc10	vsm	rele50	song
0.2994	0.3042	0.3148	0.3991	0.4803

Query Set = First Third				
roc10	roc5	vsm	rele50	song
0.2176	0.2220	0.2276	0.3018	0.4181

Query Set = Second Third				
roc5	roc10	vsm	rele50	song
0.2629	0.2783	0.3076	0.4244	0.5847

Query Set = Last Third				
roc 5	roc10	vsm	rele50	song
0.2603	0.2675	0.2887	0.4261	0.6212

Table C.11: R–Precision results for relevant set size = 11 to 15 – In these trials VSM does not differ significantly from roc10. Roc5 does not differ significantly from VSM except in the trial that samples queries from the second third. Language modeling is significantly better than the other algorithms.

Query Set = All				
roc5	roc10	vsm	rele50	song
0.2074	0.2076	0.2148	0.2902	0.3748

Query Set = First Third				
roc10	roc5	vsm	rele50	song
0.1983	0.2030	0.2030	0.2767	0.3834

Query Set = Second Third				
roc10	roc5	vsm	rele50	song
0.1894	0.1946	0.2347	0.3677	0.5091

Query Set = Last Third				
roc5	roc10	vsm	rele50	song
0.2040	0.2109	0.2605	0.3890	0.5309

Table C.12: R–Precision results for relevant set size = 16 to 20 – Here VSM is only significantly better than roc5 and roc10 for the trials that sample from the second and last third of the generated query sets. Language modeling is significantly better than the other algorithms.

Query Set = All				
vsm	roc5	roc10	rele50	song
0.2328	0.2400	0.2425	0.3069	0.3893

Query Set = First Third				
vsm	rele50	roc10	roc5	song
0.2883	0.3009	0.3011	0.3012	0.4214

Query Set = Second Third				
roc5	vsm	roc10	rele50	song
0.3079	0.3121	0.3140	0.3383	0.5154

Query Set = Last Third				
roc10	roc5	vsm	rele50	song
0.3116	0.3153	0.3223	0.3683	0.5419

Table C.13: R–Precision results for relevant set size = 21 to 50 – VSM, roc5, and roc10 do not differ significantly in these trials. Language modeling however is significantly better than the other algorithms.

Query Set = All				
vsm	roc10	roc5	rele50	song
0.3130	0.3226	0.3229	0.3230	0.4506

Query Set = First Third				
vsm	rele50	roc10	roc5	song
0.3240	0.3254	0.3363	0.3392	0.4389

Query Set = Second Third				
roc5	rele50	roc10	vsm	song
0.3077	0.3096	0.3121	0.3195	0.4929

Query Set = Last Third				
roc5	roc10	vsm	rele50	song
0.3099	0.3152	0.3256	0.3499	0.5174

Table C.14: R–Precision results for relevant set size = 51 to 400 – VSM, roc5, and roc10 are not significantly different. Language modeling is significantly better than the other algorithms.