RETRIEVAL PERFORMANCE PREDICTION AND DOCUMENT QUALITY

A Dissertation Presented

by

YUN ZHOU

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

September 2007

Computer Science

RETRIEVAL PERFORMANCE PREDICTION AND DOCUMENT QUALITY

A Dissertation Presented

by

YUN ZHOU

Approved as to style and content by:

W. Bruce Croft, Chair

James Allan, Member

David Jensen, Member

David Schmidt, Member

Andrew Barto, Department Chair
Computer Science

DEDICATION

To my parents, my wife and my daughter

ACKNOWLEDGMENTS

First of all, I would like to thank my advisor W. Bruce Croft, who taught me how to conduct high-quality research, and whose advice and insightful comments on my work are invaluable. I also would like to thank James Allan, David Jensen and David Schmidt for serving on my committee. I was fortunate to be a member of CIIR, one of the top IR research labs in the world. I enjoyed interesting and useful conversations with my fellow CIIR students: Jiwoon Jeon, Vanessa Murdock, Trevor Strohman ,Ben Carterette, Mark Smucker, Giridhar Kumaran, Xiaoyong Liu, Xing Wei, Xing Yi, Ao Feng and Shaolei Feng . In particular, I would like to thank Donald Metzler for his patient programming support and helpful comments on my work. I would like to thank Steve Cronen-Townsend, a former post-doctor in CIIR, for his help in my early years of Ph.D study. I would like to thank, Kate Moruzzi, the CIIR secretary, for her help over years. Finally, this thesis would not have finished without the support and encouragement from my wife Hairong Wu.

This work was supported in part by the Center for Intelligent Information Retrieval, in part by NSF grant #CNS-0454018 , in part by the Defense Advanced Research Projects Agency (DARPA) under contract number HR0011-06-C-0023,in part by NSF grant #IIS-9907018 ,   in part by The Central Intelligence Agency and the National Science Foundation under NSF grant #EIA-9983215, in part by SPAWARSYSCEN-SD grant number N66001-02-1-8903 and N66001-99-1-8912, in part by NSF grant number DUE-0226144 , and in part by Advanced Research and Development Activity and NSF grant #CCF-0205575. Any opinions, findings and conclusions or recommendations expressed in this material are the author's and do not necessarily reflect those of the sponsors.

ABSTRACT

RETRIEVAL PERFORMANCE PREDICTION AND DOCUMENT QUALITY

SEPTEMBER 2007

YUN ZHOU
Bachelor of Engineer, WUHAN UNIVERSITY OF TECHNOLOGY

Master of Science, UNIVERSITY OF MASSACHUSETTS AMHERST

Ph.D., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor W. Bruce Croft

The ability to predict retrieval performance has potential applications in many important IR (Information Retrieval) areas. In this thesis, we study the problem of predicting retrieval quality at the granularity of both the retrieved document set as a whole and individual retrieved documents. At the level of ranked lists of documents, we propose several novel prediction models that capture different aspects of the retrieval process that have a major impact on retrieval effectiveness. These techniques make performance prediction both effective and efficient in various retrieval settings including a Web search environment. As an application, we also provide a framework to address the problem of query expansion prediction. At the level of documents, we predict the quality of documents in the context of Web ad-hoc retrieval. We explore document features that are predictive of quality. Furthermore, we propose a document quality language model to improve retrieval effectiveness by incorporating quality information.

TABLE OF CONTENTS

Page

ACKNOWLEDGMENTS v

ABSTRACT vii

LIST OF TABLES x

LIST OF FIGURES xiii

CHAPTER

1. INTRODUCTION 1

   1.1 Retrieval Performance Prediction 1
      1.1.1 Problem Overview 1
      1.1.2 Motivation 5
      1.1.3 Challenges 10
   1.2 Document Quality 14
   1.3 Contributions 16
      1.3.1 Performance Prediction 16
      1.3.2 Document Quality 17
   1.4 Thesis Organization 17

2. RELATED WORK 18

   2.1 Query Performance Prediction 18
   2.2 Selective Query Expansion 23
   2.3 Document Quality Prediction for Improving Retrieval Performance 23

3. PERFORMANCE PREDICTION MODELS BASED ON RANDKED LISTS 25

   3.1 Clarity Score 26
   3.2 Ranking Robustness 31
      3.2.1 Information Retrieval on Noisy Data 32
      3.2.2 Robustness Score: Measuring Ranking Robustness 34
      3.2.3 First Rank Change 43
   3.3 Query Feedback 44
   3.4 Weighted Information Gain 48
   3.5 Summary of Performance Prediction Techniques 55
   3.6 Model Comparison for Query Expansion Prediction 58

4. EVALUATION OF PREDICTION MODELS 62

4.1 Query Performance Prediction in Traditional TREC Settings 62
4.1.1 Experimental Setup 62
4.1.2 Experimental Results 66
4.2 Query Performance Prediction in Web Search Environments 74
4.2.1 Content-based Queries 75
4.2.2 Named-Page (NP) Finding Queries 80
4.2.3 Unknown Query Types 84
4.2.4 Prediction on Realistic Web Data 90
4.3 Combination of performance predictors 94
4.4 Model Comparison for Query Expansion Prediction 95
4.5 Summary 99

5. IMPLEMENTATION OF PREDICTION MODELS 101

6. PREDICTING DOCUMENT QUALITY FOR WEB AD-HOC RETRIEVAL ... 105

6.1 Overview 105
6.2 Quality Metrics 106
6.2.1 Collection-document Distance 106
6.2.2 Information-to-noise ratio 108
6.3 Predicting Document Quality 109
6.3.1. Training data 110
6.3.2 Naïve Bayes Classifier 112
6.4 Document Quality Language Model 114
6.5 Results 114
6.5.1 Results for Precision at Top Ranks 115
6.5.2 Mean Average Precision Results 118
6.5.3 MRR Results 119
6.6 Query Analysis 119

7. CONCLUSION AND FUTURE WORK 124

7.1 Conclusions 124
7.2 Future Work 125

Appendix A : RERETRIEVAL PERFORMANCE MEASURES 129

Bibliography 131

LIST OF TABLES

Table Page

Table 3.1 : Top 5 terms compressed from the ranked list in response to query "Cruise ship damage sea life" 47

Table 3.2: Top 5 terms compressed from the ranked list in response to query "prostate cancer treatments" 47

Table 3.3 : Comparison of Prediction Techniques 55

Table 4.1: Summary of test collections used in Section 4.1 63

Table 4.2: Pearson's correlation coefficient for correlation with average precision, for clarity score(Clarity), robustness score (Robustness), Weighted Information Gain (WIG) and Query Feedback(QF). Bold cases mean the results are statistically significant at the 0.05 level. Stars represent the strongest correlation for that collection. 67

Table 4.3 : Kendall's correlation coefficient for correlation with average precision, for clarity score(Clarity), robustness score(Robustness), Weighted Information Gain (WIG) and Query Feedback(QF). Bold cases mean the results are statistically significant at the 0.05 level. Stars represent the strongest correlation for that collection. 67

Table 4.4 : Kendall's correlation coefficient for correlation with average precision, for clarity score(Clarity), robustness score(Robustness), Weighted Information Gain (WIG) and Query Feedback(QF). Bold cases mean the results are statistically significant at the 0.05 level. Stars represent the strongest correlation for that collection. Queries are the *description* part of TREC topics. 70

Table 4.5 : Comparison to other techniques. The reported numbers are Kendall's correlation coefficients for correlation with average precision on the Robust 2004 Track for robustness score, the overlap method [6] and the document perturbation method [29]. 73

Table 4.6 : Summary of data sets for content-based queries 75

Table 4.7 : Pearson's correlation coefficients for correlation with average precision on the Terabyte Tracks (ad-hoc) for clarity score, robustness score, the JSD-based method(we directly cites the score reported in [8]), query feedback(QF) and WIG. Bold cases mean the results are statistically significant at the 0.05 level. 78

Table 4.8 : Summary of data sets for named-page finding queries 81

Table 4.9 : Pearson's correlation coefficients for correlation with reciprocal ranks on the Terabyte Tracks (NP) for clarity score, robustness score, WIG, the first rank change (FRC). Bold cases mean the results are statistically significant at the 0.05 level. 82

Table 4.10 : Comparison of Pearson's correlation coefficients for correlation with retrieval performance under four possibilities on the Terabyte Tracks (NP). Bold cases mean the results are statistically significant at the 0.05 level. 86

Table 4.11: Comparison of prediction accuracy for five strategies in the mixed-query situation. Two ways to sample a query from the pool: (1) the sampled query is content-based with the probability p=0.6. (that is, the query is NP with probability 0.4) (2) set the probability p=0.4. 88

Table 4.12 : Division of test queries into three groups based on the number of clicked results 93

Table 4.13 : Distributions of WIG scores for Group A, B and C 93

Table 4.14 : Summary of test collections for query expansion prediction 96

Table 4.15 : Mean Average Precision 97

Table 6.1 : Distribution of relevant documents in the training data 111

Table 6.2: Summary of test collections 115

Table 6.3: Precision on the GOV2 collection. "Pos" means result is better than the baseline, "Neg" means result is worse than the baseline, "Eq" means result is the same as the baseline 116

Table 6.4: Precision on the WT2Gcollection 116

Table 6.5: Precision on the WT10G collection 117

Table 6.6: MAP on the three test collections 118

Table 6.7 : MRR on the three test collections... 119

Table 6.8 : MAP on the two types of queries 123

LIST OF FIGURES

Figure Page

Figure 3.1 : Clarity Score Calculation 27

Figure 3.2 : Term clarity score contributions for the top terms for two same-topic queries 31

Figure 3.3 : Robustness Score Calculation 36

Figure 3.4: Pseudo-code for computing FRC 43

Figure 4.5 : Average precision versus robustness scores for the 249 title queries from the Robust 2004 Track........ 68

Figure 4.2 Comparison of prediction accuracy between title queries and description queries 70

Figure 4.3: WIG score versus average precision for the 100 title ad-hoc queries from the Terabyte Tracks of 2004 and 2005 78

Figure 4.4: Distribution of robustness scores for NP and CB queries. The NP queries are the 252 NP topics from the 2005 Terabyte Track. The content-based queries are the 150 ad-hoc title from the Terabyte Tracks 2004-2006. The probability distributions are estimated by the Kernel density estimation method. 87

Figure 4.5 a scatter plot of model comparison scores versus delta average precision for each of the 250 queries of Robust 2004.The delta average precision is the value with relevance retrieval(REL) minus query-likelihood retrieval(QL). 97

Figure 6.1: Distribution of CDD values for low and high quality documents. 108

Figure 6.2: Distribution of information-to-noise ratios for low and high quality documents. 109

Figure 6.3: “Low quality” document retrieved in response to the query *controlling type II diabetes*. 110

CHAPTER 1

INTRODUCTION

1.1 Retrieval Performance Prediction

1.1.1 Problem Overview

In a typical retrieval system, a user forms a query according to her information need and a number of documents (usually in the form of a ranked list) are presented to the user by the retrieval system in response to the query. However, simply showing the user the results which are the best the system can provide does not mean these documents are relevant. It is important to know how effective the retrieval is. Although the user himself can judge the results, in many cases this is not practical because the user will be burdened with reading a nontrivial amount of information. Therefore, we desire an automatic way of assessing the quality of retrieval results.

In this thesis, the problem of predicting retrieval performance refers to the process of estimating the quality of the output of a text retrieval system in response to a particular information need without any relevance judgments.

User queries considered in this thesis can be classified by retrieval task into two categories[1]: informational or navigational [64]. Next we discuss these two retrieval tasks in detail.

Informational queries[2] are used in *ad-hoc retrieval* which is the task of finding a number of documents that are relevant to a particular information need. For example, "Type II diabetes" is a query of the ad-hoc retrieval task. The user who issued this query wants to acquire information on this disease. In other words, she wants to find documents relevant to a particular topic (that is, "Type II diabetes" in our example). The ad-hoc retrieval task has been used as the basis for the evaluation of retrieval models since the 1960s, but it was given the name "ad hoc" first in the TREC (Text REtrieval Conference) evaluations [10]. Since this task is based on topical relevance, that is, the semantic similarity between a given document and a query formulated by the user to express her information need, it is sometimes referred as "topic relevance task" in IR(Information Retrieval) literature. Average Precision (please refer to Appendix A for details) is often adopted for the evaluation of this task.

In response to navigational queries which are typically used in a Web search environment, a new task called *Named-Page (NP) finding* task was introduced into TREC several years ago. This is a navigational task since the purpose of this task is to

---

[1] Broder [64] divided web queries into three types: informational, navigational and transactional. In this thesis, we do not consider transactional queries, since so far there has been no widely accepted test data for such queries.

[2] In this thesis, we also call them as content-based queries and use the two terms interchangeably.

find a particular Web page, given a query describing it by name. This type of search is sometimes referred as “known-item” search in classic IR and information science. For example, “TREC proceedings” (TREC topic # NP973) is a NP query submitted by a user who is looking for the TREC publication page that contains TREC proceedings. One fundamental difference between this task and the ad-hoc retrieval task is that with respect to the NP task the user knows the document she searches for, while unknown documents may satisfy the user’s information need in the case of the ad-hoc retrieval task. Accordingly, generally there is only one correct answer for a NP query, as opposed to multiple relevant documents for a content-based (informational) query. Regarding evaluation, this task is often evaluated with MRR (mean reciprocal rank) or precision at a given cutoff rank K for low values of K (please refer to Appendix A for details on these two measures), since the user is assumed to be interested in only one result and prefer it to be ranked as high as possible.

Performance prediction is closely related to the task of system evaluation in information retrieval which refers to measuring the effectiveness of IR systems. System evaluation usually needs a test collection which typically consists of three parts: a test document collection, a test set of information needs and a set of relevance judgments. Examples are the classic CRANFIELD collection and the widely used TREC test collections. Though estimation of search quality is the common problem in both tasks, they differ significantly in a few ways.

First of all, system evaluation usually needs relevance judgments while performance prediction does not. Though the use of relevance judgments surely leads to more accurate estimation compared to zero-judgment, the high cost of producing relevance judgments precludes us from performing evaluation on a large scale. Furthermore, there are situations when obtaining relevance judgments is out of the question. For example, it is normal for a web search engine to receive millions of queries per day and providing relevance judgments for all of them or even a small fraction of them is practically impossible. Therefore, in the situations when relevance judgments are impractical to acquire, performance prediction offers an alternative way to provide valuable information on search effectiveness.

Additionally, system evaluation, as its name suggests, is more system-oriented rather than user-oriented in the sense that only a number of carefully-selected topics with relevance judgments are adopted for evaluation. In other words, one system that is found to perform well on a test collection using one set of topics may not perform well when the test topics have changed. It is a well-known fact that retrieval system performance is highly topic-dependent. Moreover, relevance judgments are made according to the stated information needs instead of actual user queries. Last, systems are measured by the *average* retrieval performance on a topic set. In short, an individual user may not acquire much information from the results of system evaluation about the performance of the retrieval in response to her specific query. In comparison, performance prediction focuses on the effectiveness of individual retrievals and accordingly is more user-oriented.

Furthermore, performance prediction can contribute to improving the efficiency of system evaluation. For example, the number of judgments required for reliably evaluating two retrieval systems may be reduced when the predicted retrieval performance of the two systems is properly incorporated.

1.1.2 Motivation

Compared to the long history of developing sophisticated retrieval models for improving performance in IR, research on predicting performance is still in its early stage. However, researchers have started to realize the importance of the prediction problem and a number of new methods have been proposed for prediction recently [1]. The ability to predict performance has the potential of a fundamental impact both on the retrieval system and the user.

From the perspective of a retrieval system, performance prediction is the first step at solving the crucial problem of retrieval consistency. It is important for an operational retrieval system to return at least acceptable results in response to most requests. As we stated before, current retrieval systems are evaluated by the *average* effectiveness on a fixed set of topics. However, an individual user may not benefit from an improvement on average performance when her information need is not covered by the topics used during the evaluation. To make matters worse, it is not unusual in IR for a given technique to improve average performance at the expense of performance on individual queries. For example, it is well-known that the query expansion technique can improve average retrieval performance. It has not been used in many operational systems because of the fact that it can greatly degrade the performance of some individual queries. Although failures on a small number of topics may not have a significant effect on average performance, users who are interested in these topics are unlikely to be tolerant of this kind of deficiency. A reliable system that always produces acceptable retrieval performance is more preferred by users than another system that works extremely well on a number of topics but occasionally makes terrible mistakes. To improve the consistency of retrieval systems, we first need to distinguish poorly-performing topics by performance prediction techniques (the strategy of picking up poorly-performing topics by relevance judgments is not practical and useful in most cases). The important role of performance prediction in improving retrieval consistency has been recognized by the IR community. For example, in 2003, the Robust Track [2] was proposed by TREC which addresses the problem of enhancing the retrieval of poorly-performing queries. As the first footprint in finding a solution to this problem, the Robust Track requires systems to rank the topics by predicted effectiveness to investigate the capabilities of systems to detect hard topics [2].

Identifying low performance queries is not the only way that performance prediction can contribute to retrieval systems. With feedback from accurate performance prediction, retrieval systems are equipped with the ability to self-diagnose and can be adjusted to the characteristics of individual information requests. Performing query-centered processing is a very desirable function for ad hoc retrieval systems. One way to do query-centered processing is to selectively use IR models based on performance prediction for the query. For example, in terms of the query expansion technique mentioned above, if we could determine in advance when the technique would fail by utilizing performance prediction, we would selectively apply this technique on a per query basis[3]. In fact, the Reliable Information Access (RIA) workshop organized by NIST in 2003 manually analyzed the causes of retrieval failure of several IR systems on several queries [3]. The primary goal of the workshop was to understand the roles of both system-related factors and query- related factor in retrieval failure. One of the major conclusions from their analysis was that if a system realizes the problem associated with a given query, then current IR techniques can improve results for a majority of the poorly-performing queries [3]. This suggests that selectively applying techniques on a per-query basis can adapt a retrieval system more to users' information needs.

Other than guiding retrieval systems to selectively apply IR techniques, current retrieval models can directly take advantage of performance prediction. For example, it is well-known that setting parameters in IR models plays a crucial role in retrieval effectiveness. Even theoretically motivated models like the language models reply heavily on parameter tuning to achieve good performance [4]. Usually there are two ways to estimate model parameters. One is by human experience and the other is to automatically choose parameters using training data. In essence, both ways depend on observed data[4] to set model parameters that will be used for unseen data. However, we know that the performance of retrieval systems can vary considerably on different queries and therefore we lack confidence in whether a model that is tuned based on

[3] This problem will be addressed in section 3.5 as an application of performance prediction.

[4] Human experience usually comes from observed data.

one set of queries will still perform well on another set of entirely new queries. If we have a reliable predictor of retrieval performance, we can either directly maximize the predicted performance or at least acquire useful information from the predicted performance when tuning model parameters.

On the other hand, from the perspective of a user, performance prediction provides valuable feedback that can be used to direct a search. For example, when the retrieved documents are estimated to be of low quality, the user may rephrase his query or be more willing to cooperate with the system to improve retrieval effectiveness, such as providing relevance feedback. With the help of prediction, the user can quickly form a good query to acquire satisfying results for her information need. Otherwise, the user must spend time reading the returned documents to rewrite the query when the results for the initial query are not satisfactory. Note that formulating a well-defined information request is not an easy task for an ordinary user, as the user may be unaware of the inherent ambiguity in her query which usually results in a poor retrieval performance. An example is the one term query “apple” that has at least the following three possible meanings: a kind of fruit called apple, the Apple computer, the Big Apple (the nickname of New York City). Even what an experienced user believes to be a well-defined query may, in fact, perform poorly depending on the system and the data. For instance, suppose a user interested in the development of an application of ocean remote sensing issues the query “ocean remote sensing” which seems a clearly-defined query. However, according to the results reported in [5], all the retrieval systems in [5] fail to recognize the importance of the aspect of “ocean” and there is little chance for the user to forecast this kind of failure.

In addition to being a useful tool from the perspective of both user and system, performance prediction can play an important role in user-system interaction. As we stated above, even an expert may find it difficult to formulate an effective query that accurately reflects her information need, since much information about the retrieval system, is unavailable to the user. For an ambiguous query issued, instead of shifting the burden of query disambiguation to the retrieval system, an alternative way is to invoke use-system interaction and gather more information from the user. One example of user-system interaction is *query refinement* that allows the user to interactively specify her information need by selecting new terms suggested by the system.

One important issue in user-system interaction is when to apply it, since interaction is not necessary for every query. In fact, selecting a well-formed query for interaction can not only degrade the user experience but also make the retrieval system inefficient. One possible solution to this problem is applying query performance prediction. For example, we can invoke interaction only when the predicted query performance is below some threshold. Some researchers have started to work on incorporating performance prediction in interaction [63]. We believe that integrating performance prediction into user-system interaction can results in the design of more intelligent and user-friendly information retrieval systems.

Retrieval performance prediction also has a natural application to the field of distributed information retrieval where the retrieval process is performed over multiple databases. Performance predictors that run on each database will provide helpful information for database selection and merging results from all databases, which are two fundamental problems in distributed IR. Elad et al [6] have shown some early work in this direction.

Performance prediction, as we described above, has potential applications in many important IR areas such as system evaluation, user-system interaction, ranking strategies and distributed IR. Furthermore, to develop a good performance predictor, we require a deep understanding of the strength and weakness of retrieval techniques on a per query basis. We also need to understand the relationship of the query to the system as a whole. These not only enable us to comprehend the fundamentals behind the observed performance, but also to provide query-centered processing either by selectively applying multiple IR models or by adjusting each model on a per-query basis. This will be appreciated by individual users and represents a novel approach to IR research over the “single fixed model for all queries” which has prevailed so far.

1.1.3 Challenges

The major difficulty in performance prediction comes from the fact that many factors have an impact on retrieval performance. Each factor affects performance to a different degree and the overall effect is hard to predict accurately. Typically, the following three aspects are most responsible for retrieval effectiveness.

(1) Query Quality

Some queries, like the query “apple” described above, are ambiguous and it is not easy for a retrieval system to understand the true information need of a user without more information beyond the query itself. Sometimes there is little ambiguity in every single query term, but the whole query is still obscure. Such an example is TREC topic 413 “what are new methods of producing steel”. The interpretation of “new methods” is unclear even for humans. As we can see, the type of failure is difficult for performance predictors to catch since it needs natural language understanding.

(2) Collection Characteristics

The nature of collections can affect performance in unobvious ways. For example, intuitively the more relevant documents that exist in a collection for a given query, the higher the retrieval precision will be. However, Giambattista et al. reported [20] that they observed a negative correlation between the number of relevant documents and the precision (measured by mean average precision or precision at top 10 documents) on the TREC disk 4 and 5 using 100 TREC queries. Their explanation is that a small number of relevant documents suggest that the query is specific and is relatively easy for the system to retrieve. Another example is that if a collection contains quite a few non-relevant documents that are similar to some relevant documents, the retrieval performance will be lower.

(3) Retrieval Model

A typical retrieval model consists of three parts: query representation, document representation and a retrieval function that explicitly or implicitly estimate the probability of relevance based on the query and document representation. The three parts interact with each other in a complicated way and every part can exert a significant impact on retrieval performance either alone or with the others. Even implementation details such as stemming may result in failure for a given query. For example, as reported in [5], being unable to stem “Antarctica” to “Antarctic” caused the failure of TREC topic 353 (“Antarctica exploration”).

Usually a combination of factors affects performance and it is difficult to separate one factor from the others, making the prediction problem more complicated. For instance, suppose a user who is a soccer fan issues the query “World Cup” against a collection to search for articles about World Cup soccer. If the collection happens to contain a number of documents about World Cup chess which can be highly ranked by the system, the performance will be low. If there is no other kind of “World Cup ” in the collection, the user’s query will be effective. In this example, we can not simply attribute the problem to query ambiguity or collection characteristics. It is the relationship of the query to the collection that causes the problem. Considering the fact that the user’s query, the collection and the retrieval model act as a whole on the overall retrieval performance, we think that any predictor that takes only one aspect into account is not likely to be accurate.

In addition to the above challenges, the advent of the Web further complicates the prediction task. Due to the popularity and influence of the Web, next we discuss some of the major challenges to prediction posed by web search environments.

First, web collections, which are much larger than conventional test collections, include a variety of documents that are different in many aspects such as quality and style. Prediction techniques can be vulnerable to these characteristics of web collections. For example, some state-of-the-art prediction techniques perform significantly worse on a large web collection compared to other non-web collections [7,8].

Furthermore, web search goes beyond the scope of the ad-hoc retrieval task based on topical relevance. For example, the Named-Page (NP) finding task, which is a navigational task, is also popular in web retrieval. Query performance prediction for the NP task is still necessary since NP retrieval performance is far from perfect. In fact, according to the report on the NP task of the 2005 Terabyte Track [9], about 40% of the test queries perform poorly (no correct answer in the first 10 search results) even in the best run from the top group. To our knowledge, little research has explicitly addressed the problem of NP-query performance prediction. Prediction models devised for content-based queries will be less effective for NP queries considering the fundamental differences between the two.

Third, in real-world web search environments, user queries are usually a mixture of different types and prior knowledge about the type of each query is generally unavailable. The mixed-query situation raises new problems for query performance prediction. For instance, we may need to incorporate a query classifier into prediction models. Despite these problems, the ability to handle this situation is a crucial step towards turning query performance prediction from an interesting research topic into a practical tool for web retrieval.

1.2 Document Quality

In the problem of retrieval performance prediction discussed above, we predict retrieval quality at the granularity of the retrieved document set as a whole. Here we are interested in predicting the quality of individual documents within the context of Web ad-hoc retrieval. Traditionally, retrieval models for ad-hoc retrieval have focused on capturing topics through word distributions. For example, in the query likelihood language modeling approach [11], documents are ranked by the probability that their underlying language model can "generate" the query. Other factors relating to document content, such as the quality or genre of the text, have had very little impact. However, with the advent of the Web and test collections derived from the Web, it is clear that these other content-related document properties are much more important. In particular, due to the relative simplicity of generating and publishing web documents, the quality and style of web documents varies much more widely than the newswire-based TREC test collections. Web pages vary in quality from well-written articles to pages with very little or even no real content.

Empirical studies play an important role in IR research and many successful information retrieval (IR) systems heavily rely on the empirical tuning of model parameters. Therefore the performance of IR models typically has a close relationship with the characteristics of test collections. When the characteristics of the test collections change, as with the introduction of large Web-based collections, problems with the retrieval model can be exposed. For example, the query ‘artificial intelligence’ (TREC topic 741), when used to retrieve Web pages (using the query likelihood model) from the TREC GOV2 web collection [10], ranks lists of AI conferences or papers at the top, although these do not directly describe artificial intelligence at all. They are highly ranked only because the two query terms occur many times in the documents. In other words, the retrieved documents are topically relevant but are not the right type of document. In this paper, we consider this type of retrieval failure (and others described later) to be related to document “quality” and propose methods for allowing quality to influence ranking.

There has been a considerable amount of research related to Web page quality based on links. PageRank[12] and HITS[13] are two of the best-known algorithms for link structure analysis. The basic idea behind these link-based models is that a page to which many documents link is popular and therefore is likely to be of high quality. While link-based methods are clearly effective at estimating popularity, this is only one aspect of document quality. Link information has been shown to be valuable for the home-page and named-page finding TREC tasks [14,15], but participants in recent TREC web tracks [16,17,18] consistently reported that there is no conclusive benefit from the use of link information for the ad hoc task (sometimes called “content-based retrieval”). In fact, incorporating link information can sometimes even hurt retrieval performance [16,17,18].

A major goal of this study is improving the performance of Web ad-hoc retrieval by exploiting document quality information. Specifically, we are interesting in these two problems: (1) how to measure quality, and (2) how to incorporate quality information into a retrieval model.

1.3 Contributions

1.3.1 Performance Prediction

The work in this thesis helped to define the area of retrieval performance prediction. The experiments are the most comprehensive yet done in terms of test collections and query types used. These experiments show that satisfactory prediction accuracy can be achieved across a variety of search scenarios.

In particular, one of our performance prediction techniques, WIG (weighted information gain), demonstrates superiority over most of the others when it comes to Web search. WIG provides a uniform framework to deal with both content-based and named-page finding queries. To our knowledge, most past work has only considered content-based queries. In addition, with the help of an automatic query classifier, WIG offers a practical solution to predicting mixed-query performance, which is a crucial step towards turning query performance prediction from an interesting research topic into a practical tool for Web search. Our experiment on realistic Web data collected from a commercial Web search engine shows a tendency that high WIG scores predict more clicks on search results. In addition, WIG can be implemented very efficiently.

As an application, we study the task of predicting the performance difference between an expanded retrieval and an unexpanded retrieval for a given query. We provide a framework called *model comparison* for this task.

1.3.2 Document Quality

Document quality can be viewed as predicting retrieval quality at the level of individual documents. This work offers a number of contributions. First, we propose a new document quality metric that was found to be helpful for identifying low quality documents. Second, our results show that the document quality model proposed by us can improve accuracy for Web ad hoc retrieval. Third, our query analysis provides some interesting insights on the relationship between document quality and relevance.

1.4 Thesis Organization

The rest of this thesis is organized as follows. Chapter 2 describes related work. In Chapter 3, we introduce several models for performance prediction. We also approach the task of query expansion prediction in this chapter. Experimental results for prediction are shown in Chapter 4. In Chapter 5, we discuss implementation of prediction models. Chapter 6 addresses the problem of document quality. Conclusions and future work are presented in Chapter 7.

CHAPTER 2

RELATED WORK

2.1 Query Performance Prediction

When we predict the quality of the retrieved documents for a given query, we call it query performance prediction. Prediction of query performance has long been of interest in information retrieval and has been investigated under different names such as query-difficulty or query-ambiguity [20,21]. Our work, the clarity score method originally proposed in [19] for prediction, demonstrated some of the first success at addressing this task.

Recently, a number of prediction methods have been tried since the introduction of the TREC Robust Track in 2003. In the Robust Track systems are required to rank the queries by predicted performance, with the goal of utilizing the prediction capability to do query-specific processing. Generally speaking, these methods extract features of retrieval and compute the performance score for each query by using the features to estimate the query performance. One way to measure the quality of the performance prediction methods is to compare the rankings of queries based on their actual precision (such as MAP) with the rankings of the same queries ranked by their performance scores (that is, predicted precision). Based on whether retrieval results are needed when computing the performance score, these methods can be classified into two groups: pre-retrieval approaches and post-retrieval approaches.

Category I: Pre-retrieval approaches

In this category, performance predictors do not rely on the retrieved document set. The efficiency of this kind of predictor is often high since the performance score can be computed prior to the retrieval process. However, regarding prediction accuracy, these predictors generally have a low performance since many factors related to retrieval effectiveness are ignored.

Some researchers have used IDF-related (inverse document frequency) features as predictors. For example, Tomlinson et al. [22] adopted the weighted average IDF of the query terms for predicting. He and Ounis [23] proposed a predictor based on the standard deviation of the IDF of the query terms. Plachouras [24] represented the quality of a query term by Kwok's inverse collection term frequency. The above IDF-based predictors showed some moderate correlation with query performance.

Diaz and Jones [25] have tried time features for prediction. They found that although they are not highly correlated to performance, using these time features together with clarity scores improves prediction accuracy.

Kwok et al. [26] built a query predictor using support vector regression. For features, they chose the best three terms in each query and used their log document frequency and their corresponding frequencies in the query. They observed a small correlation between predicted and actual query performance.

He and Ounis [23] proposed the notion of query scope for performance prediction, which is quantified as the percentage of documents that contain at least one query term in the collection. No strong correlation with retrieval performance was observed.

Category II: Post-retrieval approaches

In this category, predictors make use of retrieved results in a variety of ways. All of our prediction models fall into this category. Generally speaking, techniques in this category provide better prediction accuracy compared to those in category I. However, computational efficiency can be an issue for many of these techniques. Fortunately, some of our techniques can achieve satisfactory prediction accuracy without sacrificing efficiency.

Bernstein et al. [27] estimate the prior probability of each document that will be retrieved by the retrieval system. For a given query, they compare the ranking of documents based on the prior probabilities to the ranking of documents returned from the retrieval system. They hypothesize that if the two rankings are similar, the query will be difficult since the query does not have strong discriminating power. Their results show some limited indication of query performance.

Using visual features, such as titles and snippets, from a surrogate document representation of retrieved documents, Jensen et al. [28] trained a regression model with manually labeled queries to predict precision at the top 10 documents in the Web search. They reported moderate correlation with precision.

Elad Yom-Tov et al. [6] proposed a histogram-based predictor and a decision tree based predictor. The features used in their models were the document frequency of query terms and the overlap of top retrieval results between using the full query and the individual query terms. Their idea was that well-performing queries tend to agree on most of the retrieved documents. They reported promising prediction results and showed that their methods were more precise than those used in [26][24][22]. We call this technique the *overlap* method in this thesis.

A few techniques are based on measuring some characteristics of the retrieved document set to estimate performance. For example, our clarity technique measures the coherence[5] of the retrieved document set. In fact, the initial success of the clarity method has inspired a number of similar techniques. Amati [20] proposed to use the KL-divergence between a query term's frequency in the top retrieved documents and the frequency in the whole collection, which is very similar to the definition of the clarity score. He and Ounis [23] proposed a simplified version of the clarity score where the query model is estimated by the term frequency in the query. Carmel et al. [8] found that the distance measured by the Jensen-Shannon Divergence (JSD) between the retrieved document set and the collection is significantly correlated to average precision (we call it the *JSD* method in this thesis).

Vinay et al.[29] propose four measures to capture the geometry of the top retrieved documents for prediction. The most effective measure is the sensitivity to document perturbation (we call it the *document perturbation* method in this thesis), an idea somewhat similar to one of our techniques: the robustness score. Unfortunately,

[5] Broadly speaking, coherence means topical similarity among the retrieved documents. In the case of clarity score, it means the extent that the retrieved documents use the same certain terms.

their way of measuring the sensitivity does not perform equally well for short queries and prediction accuracy drops considerably when a state-of-the-art retrieval technique (like Okapi or a language modeling approach) is adopted for retrieval instead of the tf-idf weighting used in their paper [30].

Kwok et al. [31] suggest predicting query performance by retrieved document similarity. The basic idea is that when relevant documents occupy the top ranking positions, the similarity between top retrieved documents should be high, based on the assumption that relevant documents are similar to each other. While this idea is interesting, preliminary results are not promising.

Diaz [67] proposes a technique called spatial autocorrelation for performance prediction. This technique measures the degree to which the top ranked documents (for a given retrieval) receive similar scores by spatial autocorrelation of the retrieval. This approach is based on the cluster hypothesis [58]: closely-related documents tend to be relevant to the same request. A significant correlation between score consistency and retrieval performance was observed in their experiments.

One important issue we want to point out is that most work on prediction has focused on the traditional ad-hoc retrieval task where query performance is measured according to topical relevance. In fact, we know of no published work of other researchers that addresses other types of queries such as named-page finding (NP) queries, let alone a mixture of query types. Moreover, these prediction models are usually evaluated on traditional TREC document collections which typically consist of no more than one million relatively homogenous newswire articles. In this thesis, we will present techniques that are capable of dealing with different retrieval tasks and thoroughly evaluate them against a variety of test collection, including a large web collection.

2.2 Selective Query Expansion

Automatic query expansion is a well-known IR technique that has been studied extensively. For example, [32] and [33]. Although query expansion has been shown to be capable of improving average retrieval performance, one common criticism of this technique is that it can diminish the retrieval quality for some queries.

Only recently have some researchers begun to study predicting query expansion failure. Amati [20] proposed a measure similar to query length for predicting expansion. Yom-Tov et al [6] used an SVM classifier for this purpose. Research in this area is still in its infancy and no one has reported significant results in this direction. The key problem is that we lack the understanding of this technique on a per-query basis.

2.3 Document Quality Prediction for Improving Retrieval Performance

The research on link-based approaches to the popularity aspect of quality, such as PageRank[12] and HITS[13], has already been mentioned earlier. There have been many attempts to combine link information with content-based IR approaches to improve Web ad hoc retrieval performance [14,15,16,34,35]. However, no consistent and conclusive improvements have been demonstrated. In this thesis, we focus on using content-based features rather than links to estimate document quality.

Other related work uses prior probabilities to improve IR based language modeling. The language modeling approach provides a convenient framework for incorporating prior knowledge in the form of prior probabilities. A variety of prior information, such as document length and time, has been used for ad hoc retrieval [36,37]. Kraaij et al [38] also used Web-specific features as prior knowledge for a home page retrieval system. In our approach, the prior probabilities in the language model framework are based on estimates of document quality based on content features.

There have been few attempts to directly integrate document quality into ad hoc retrieval. Zhu and Gauch [39] show that incorporating quality metrics can improve precision in a web search environment. They combine quality metrics into a vector-based algorithm in a heuristic way. The quality metrics they studied were related to currency, availability, information-to-noise ratio, authority, popularity and cohesiveness. They found information-to-noise to be possibly the most effective metric and we use this measure in our study. The other metric we use (collection-document distance) is new. One major limitation of the work is that the non-standard test collection used for evaluation is extremely small (less than 1500 documents). In fact, the test collection only comes from twenty target sites and only covers five topics. We evaluate our technique on three different Web collections that contain millions of document.

CHAPTER 3

PERFORMANCE PREDICTION MODELS BASED ON RANKED LISTS

In a typical retrieval system, the user submits a query and the system returns a ranked list of documents in response to the query. The characteristics of the ranked list provide useful information for predicting retrieval performance.

In this chapter, we describe four query performance prediction techniques that capture and quantify some properties of the ranked list. The first model, named the clarity score, is designed to measure the coherence of the ranked list. Ranking robustness, our second model, is proposed for measuring the robustness of the ranked list. Our third model, called Query Feedback (QF), measures to what extent we can restore the original query from the ranked list. Our last model, called Weighted Information Gain (WIG), measures the change in information from an imaginary state where only an average document is retrieved to a posterior state that the actual ranked list is observed.

In addition to the task of query performance prediction, we also address the problem of predicting when to apply query expansion to a particular query. We develop a method called Model Comparison (MC) that is built on clarity score-related ideas.

3.1 Clarity Score

In this section we introduce a measure of the coherence of a ranked document list called the clarity score. Here “coherence” means the extent to which top ranked documents use similar language. Therefore, the clarity score can be used to distinguish a coherent ranked list that uses similar language from an incoherent ranked list containing a huge variety of documents in terms of word-usage.

The reason we are interested in measuring coherence is that there is a relation between the coherence of a ranked list and the chances of that list containing many relevant documents. To illustrate this relation, let us consider two ranked lists of documents returned for the same given query: an incoherent one and a coherent one. In the incoherent ranked list containing documents of greatly differing word usage generally no more than one document is relevant because these top ranked documents are very different. On the other hand, in the coherent ranked list the likely options are that either many of the documents are relevant or none of them are relevant. The former option is much more likely because the document list was ranked using the query.

To measure coherence, we first build a query language model which is based on the retrieved documents using the query. This query language model represents the language usage in the ranked list in response to the query. Then the query model is compared to the collection model representing the average language usage in the collection. The KL-divergence between the query and collection language models is the *clarity score* for the query. This process is shown in Figure 3.1. To understand why this measure can estimate ranking coherence, let us think of a query whose highly ranked documents are roughly about the same topic (which means high coherence). In this case, the large probabilities of the query language model are allocated to a small number of topic terms. Therefore, the KL-divergence between the two models is high. On the other hand, if a ranked list consists of a mix of documents about different topics, we can imagine that the word probabilities in the query model would be more evenly distributed, leading to a low clarity score.

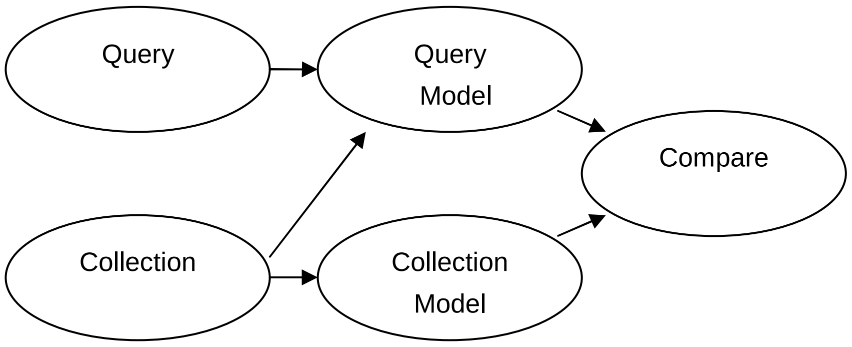


Figure 3.1 : Clarity Score Calculation

Next we discuss details regarding clarity calculation. Given query Q and collection C , the query language model $P(w|Q)$ is estimated by the relevance model proposed in [40], that is,

$$P(w|Q) = \sum_D P(w|D)P(D|Q) \qquad (3.1)$$

where w is any term and D is a document in the collection.

This can be interpreted as a weighted average of document models, $P(w|D)$, with weights given by $P(D|Q)$.

The document model $P(w \mid D)$ is given by

$$P(w \mid D) = \lambda P_{ml}(w \mid D) + (1-\lambda)P_{coll}(w) \qquad (3.2)$$

where $P_{ml}(w \mid D)$ is the relative frequency of term w in document D, $P_{coll}(w)$ is a *collection model* and is estimated by the relative frequency of the term in the collection C. λ is a smoothing parameter ranging from zero to one. Generally there are two popular ways to set λ. One is Jelinek-Mercer smoothing which sets λ to a constant value [4]. The other way is Dirichlet smoothing [4] where λ is estimated by:

$$\lambda = \frac{|D|}{|D| + \mu} \qquad (3.3)$$

where μ is a parameter called document prior and |D| is the length of document D.

For the weight $P(D \mid Q)$ in Equation 3.1, we first perform query likelihood retrieval [41]. We estimates the likelihood of an individual document model generating the query as

$$P(Q \mid D) = \prod_{w \in Q} P(w \mid D) = \prod_{w \in Q} \lambda P_{ml}(w \mid D) + (1-\lambda)P_{coll}(w) \qquad (3.4)$$

and obtain $P(D \mid Q)$ by Bayesian inversion with uniform prior distribution for documents.

We find it important to estimate the document model P(w|D) in Equation 3.2 by Dirichlet smoothing. Jelink-Mercer smoothing is appropriate for mixing the document models in Equation 3.4. Our finding is consistent with what is found in relevance model retrieval [40] and Zhai and Lafferty's paper about smoothing [4].

We can see that the query model $P(w|Q)$ defined in Equation 3.1 represents the language usage in documents closely related to the query, since terms that are prominent in the query model are those occurring frequently in documents whose model are likely to generate the query.

Theoretically, the summation in Equation 3.1 is done over all documents. However, in practice this is infeasible and unnecessary. We truncate the summation at the top 500 documents. That is, we consider the top 500 documents retrieved by the query likelihood retrieval. Since $P(D|Q)$ generally drops sharply well before this cutoff, this cutoff has very little effect on clarity calculations.

Finally, the clarity score is defined as the KL-divergence (or the relative entropy) between the query and collection models

$$clarity\,score = \sum_{w \in V} P(w|Q) \log_2 \frac{P(w|Q)}{P_{coll}(w)} \qquad (3.5)$$

where V is the vocabulary of the collection.

The collection model $P_{coll}(w)$ represents the average language usage in the collection. The KL-divergence is a measure of the difference between two probability distributions. Under this scheme, a query matching documents using very generic language (on average) receives a score near zero, and a query matching documents using a certain specialized vocabulary (on average) receives a relatively high score.

Next we give two examples. Figure 3.2 shows the top 50 individual term contributions to the summation in Equation 3.5 (clarity score) for two same-topic queries. The two queries are query A: “How does computerized medical diagnosis circumvent the need to use invasive techniques?” with a clarity score of 3.53 and query B: “What are the current and future medical innovations and improvements?” with a score of 0.73. Top contributing terms are those whose probabilities most stand out in comparison to the collection model, such as “invasive” and “diagnosis” for query A. The total clarity score is the total of all bars for the appropriate query if one imagines extending the plot to include all vocabulary terms. The figure shows that the query model for query A is much more unusual than the collection model, and shows the contributions for these spikes. This is due to its highly-scoring documents using the same certain terms, what we call coherence. Query B has much lower maximum contributions and a lower total (clarity score). The fact that the medical term “enthesopathy” was one of the top contributing terms in query A’s clarity score while the top terms of query B’s language model are all fairly general is a good indicator that query A is a better performing query than query B. For Query A, “Enthesopathy” occurred in documents that had high query likelihood scores, leading to an estimate of its probability in the query model well above its collection model probability. For Query B, by contrast, the only terms that stand out are fairly general terms, indicating that P(D|Q) is less focused (peaked) on a coherent set of documents.

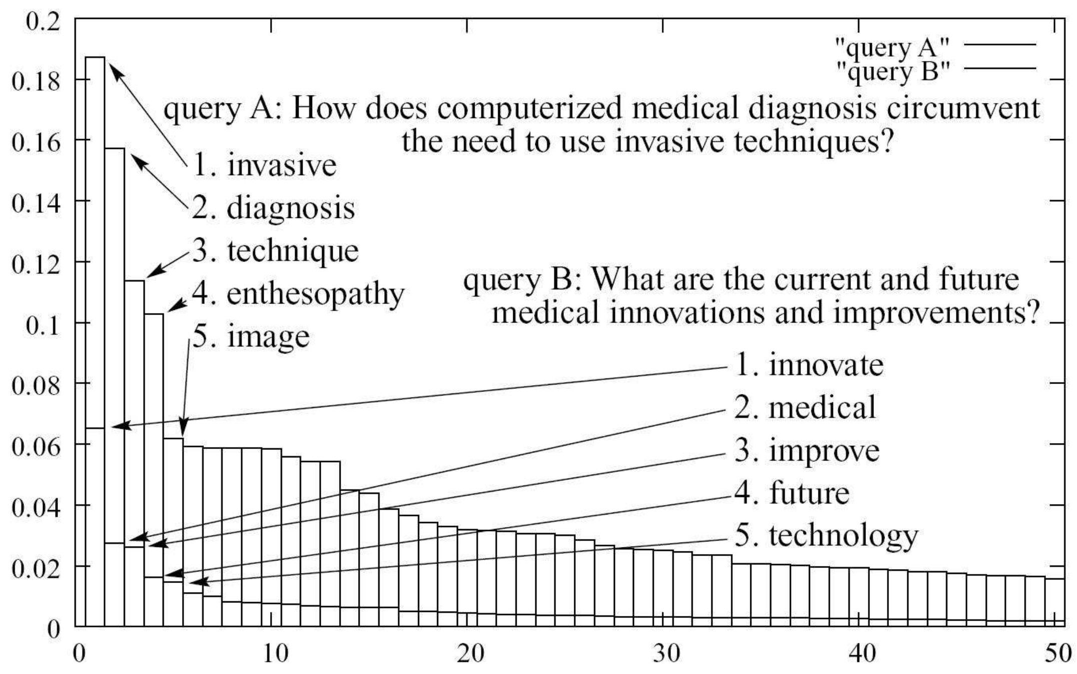


Figure 3.2 : Term clarity score contributions for the top terms for two same-topic queries

As we stated earlier in this section, the clarity score measures to what extent the top ranked document will form a coherent topic. Therefore, the use of the clarity score to predict retrieval performance is particularly appropriate for the ad-hoc retrieval task based on topic-relevance. However, for other fundamentally different retrieval tasks, such as the named-page finding task, prediction using the clarity score will consequently be much less effective.

3.2 Ranking Robustness

In this section, we describe another property of a ranked list for performance prediction: robustness. A measure called the *robustness score* is proposed to measure ranking robustness. The robustness score is primarily designed for content-based queries and is less effective for named-page finding queries. Accordingly, we develop another method called the *first rank change (FRC)* by modifying the robustness score technique to measure ranking robustness for named-page finding queries.

The notion of ranking robustness originates in the field of noisy data retrieval. We first introduce the background in noisy data retrieval that inspires our ideas.

3.2.1 Information Retrieval on Noisy Data

With regard to text document collections in information retrieval, it is often convenient to assume that the contents of the collections are clean and free of errors. With the advent of large collections of multimedia documents (such as audio or image document), techniques such as OCR (optical character recognition) or ASR (automatic speech recognition) have been widely used to extract text from multimedia archives. In the following description, the text output of a recognition process applied to multimedia documents is *noisy data* or *corrupted data* since the recognition process is error prone and brings significant levels of noise to the data. The recognition process that produces corrupted data is called *data corruption*.

One of the core problems in the field of information retrieval on corrupted data is to explore the impact of data corruption on retrieval effectiveness in order to design a ranking function that is robust to unexpected errors in corrupted data. Here a robust retrieval model means that some changes in document or collection statistics caused by data corruption do not alter the retrieval results significantly compared to retrieval on perfect documents (that is, the results of a recognition process with 100% accuracy).

A general observation about experiments on investigating the effects of data corruption is that as retrieval effectiveness improves, the ranking function becomes more robust against data corruption. For example, Lopresti and Zhou [42] explored the effectiveness of three retrieval functions on simulated OCR noisy data. They found that the ranking of the three functions with respect to retrieval effectiveness is the same as their ranking with respect to their ability to deal with simulated noise. Another example is that Singhal, Salton and Buckley [43] proposed a new robust length normalization method to alleviate the problem that the regular cosine normalization is sensitive to OCR errors. Although the original motivation for this technique was to deal with OCR data corruption, surprisingly they found that the new normalization scheme also brought significant improvements on correct text collections in comparison to the original cosine normalization. Moreover, Mittendorf [44] studied data corruption effects on retrieval and presented a theorem on ranking robustness that partially explained the phenomenon that retrieval performance on corrupted data is often correlated with the degree of resilience against noise.

The above work reveals the interesting relationship between ranking robustness and retrieval performance. Although this work was done in the context of retrieval on noisy data, clean documents in regular retrieval also contain “noise” if we interpret noise as uncertainty. Next, we will propose a framework to quantify ranking robustness.

3.2.2 Robustness Score: Measuring Ranking Robustness

As we mentioned before, the notion of ranking robustness originates in the field of noisy data retrieval, where retrieval is performed on the output of a recognition process that exacts text from multimedia archives. Ranking robustness in noisy data retrieval refers to a property of a ranked list of documents that indicates how stable the ranking is in the presence of noise brought by the recognition process. Note that clean documents also contain "noise" if we generalize the notion of noise from recognition errors to uncertainty in text documents. For example, the meaning of a document may remain the same even after adding or deleting some words. Synonymy and homonymy are another two popular examples that can bring uncertainty to clean text documents. Therefore, we can extend the notion of ranking robustness to regular ad-hoc document retrieval. In essence, ranking robustness reflects the ability of a retrieval system to handle uncertainty.

The idea of predicting retrieval performance by measuring ranking robustness is inspired by a general observation in noisy data retrieval that the degree of ranking robustness against noise is positively correlated with retrieval performance. We hypothesize that when it comes to regular ad-hoc retrieval, the positive correlation between robustness and performance still holds. Our hypothesis will be thoroughly examined in the evaluation chapter of this thesis.

Next we describe our way of measuring ranking robustness for ad-hoc retrieval. We begin by considering how to calculate ranking robustness in noisy data retrieval. If we can acquire a clean version of the corrupted data, one straightforward way is to compare a ranked document list from the corrupted collection to the corresponding ranked list from the perfect collection using the same query and ranking function. With regard to regular document retrieval, usually documents are assumed to be free of corruption. To simulate data corruption, we assume that there exists a noisy channel which is analogous to the recognition process in noisy data retrieval. Documents are corrupted after going thought the channel. One way to implement the noisy channel is to design a document model for each document (document models are distributions over words or other linguistic units). One corrupted version of the original document is one random sample from the corresponding document model.

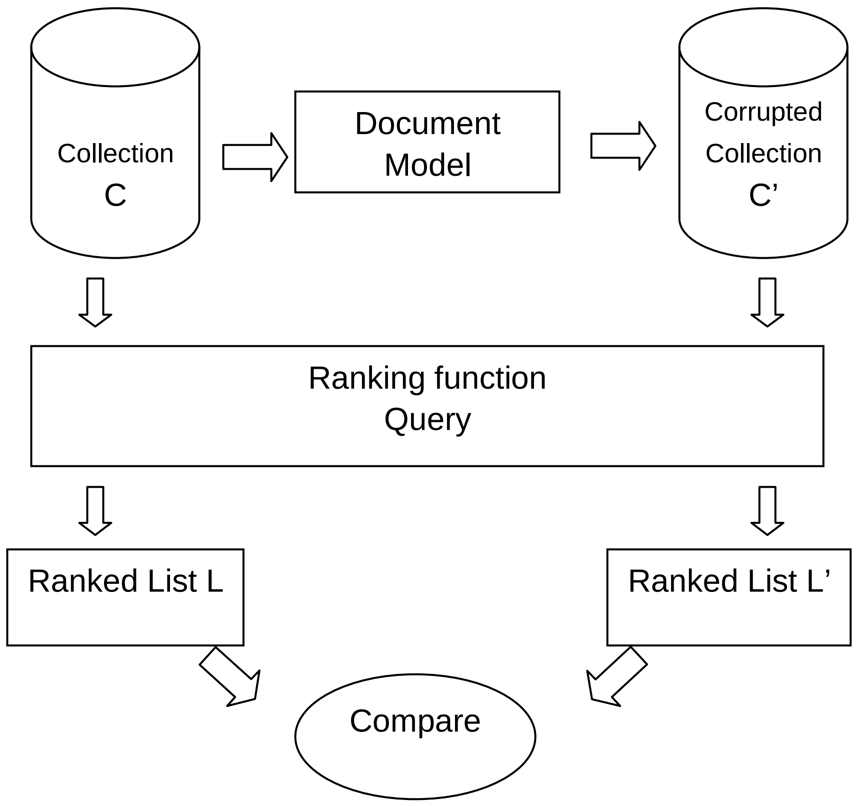


Figure 3.3 : Robustness Score Calculation

Specifically, suppose we have query *Q*, ranking function *G* and collection *C*. We generate corrupted collection *C*’ by sampling from the document models of the documents in *C*. Then we perform retrieval on both *C* and *C*’ and two ranked list *L* and *L’* are returned respectively. Finally we compute the similarity between the two rankings. Note that *L* is a fixed ranked list while *L’* is a random variable. We call the expected similarity between *L* and *L’* the robustness score and use it to measure ranking robustness. This process is illustrated in Figure 3.3.

Let us formally define the robustness score. Consider query *Q* and a document collection of M documents $C=(D_1,D_2,\ldots D_M)$. Let *V* denote the size of vocabulary, both query Q and the documents are represented as vectors of indexed term counts, that is,

$Q=(q_1,q_2,\ldots q_V)\in N^V$

$D_k=(D_{k,1},D_{k,2},\ldots D_{k,V})\in N^V$

where $D_{k,i}$ is the number of times that term i appears in document D_k and q_j is the number of times that term j appears in query Q. N denotes nonnegative integer and N^V denotes a V-dimension vector space of nonnegative integer. Under our representation, collection C is a $M\times V$ matrix with nonnegative integer entries, that is, $C\in S(M\times V)$, where $S(M\times V)$ denotes the set of a $M\times V$ matrix with nonnegative integer entries . The rows of matrix C can be viewed as a set of documents represented by V-dimension vectors.

We introduce a few definitions before we show the computation of the robustness score.

Definition 1: Retrieval Function *G(D,Q)*

retrieval function G(D,Q) maps query Q and document D into a real number, that is ,

$G(D,Q)\in R, D\in N^V, Q\in N^V$

Definition 2: Ranked List *L(Q,G,C)*

Let S_M denote the set of permutation of {1,2..M}. Ranked list $L(Q,G,C)\in S_M$ is a permutation of the documents in collection C that describes the ordering of documents by decreasing G(D,Q) where $D\in C$

***Definition 3*:** Document Model X_k and Probability Mass Function (pmf) $f_{X_k}(x)$

We assume that document $d_k, k \in [1, M]$, corresponds to document model X_k which is a V-dimension multivariate distribution and can be represented by a random vector $X_k = \{X_{k,1}, X_{k,2}, ... X_{k,i}, .. X_{k,V}\} \in N^V$, where random variable $X_{k,i}$ denotes the number of times term i occurs. The joint pmf of X_k is the function defined by $f_{X_k}(x) = f(x_1, ..., x_V) = \Pr(X_{k,1} = x_1, ..., X_{k,V} = x_V)$ where $x = (x_1, ..., x_V) \in N^V$.

***Definition 4*:** Ranking Similarity $SimRank(L_1, L_2)$

Given two ranked list $L_1(Q, G, C_1)$ and $L_2(Q, G, C_2)$, function $SimRank(L_1, L_2)$ returns a real number that measures the similarity between the two ranked lists (we assume that the documents in C_1 have one-to-one correspondence to the documents in C_1). Moreover, $SimRank(L_1,L_2)$ should be bounded.

Definition 5: Random Collection X

Given document model $X_1, \ldots X_M$, where X_k (k$\in$ [1,M]) is a V-dimension random vector, we define random collection X=(X_1,X_2,…X_M) ,that is, X is a $M \times V$ matrix whose entries consist of random nonnegative integers from some distributions. The pmf of X is the function defined by $f_X(T) = f_X(t_1, ..., t_M) = \Pr(X_1 = t_1, ..., X_M = t_M)$, where X_k denotes the k-th row of X and $t_k \in N^V$, k$\in$ [1,M].

With the above definitions, we give the definition of the robustness score.

Given query $Q \in N^V$, retrieval function G, collection $C=(D_1,D_2,\ldots D_M)\in S(M\times V)$ and random collection $X=(X_1,X_2,\ldots X_M)$, the robustness score is defined as the expected value of random variable *SimRank(L(Q,G,C),L(Q,G,X))*:

$$\begin{aligned} &\mathit{Robustness\ Score}(Q,G,C,X) = E\{\mathit{SimRank}(L(Q,G,C),L(Q,G,X))\} \\ &= \sum_{T\in S(M\times V)} \mathit{SimRank}(L(Q,G,C),L(Q,G,T))f_X(T) \qquad (3.6)\end{aligned}$$

To make Equation 3.6 feasible to calculate, we further make the following five assumptions:

(1) We assume independence between any two document models X_i and X_j, that is,

$$f_X(T) = f_X(t_1,t_2,\ldots t_M) = \prod_{k=1}^{M} \Pr(X_k = t_k) = \prod_{k=1}^{M} f_{X_k}(t_k) \qquad (3.7)$$

(2) Instead of the whole collection, only the top *J* retrieved documents in *L(Q,G,C)* and the corresponding *J* documents in *L(Q,G,X)* are used to compute the similarity between the two ranked lists. For the purpose of rank comparison, the corresponding *J* documents in *L(Q,G,X)* will shift up in rank and form a new ranked list of length *J*.

(3) The Spearman rank correlation coefficient [45] is adopted to compute the value of function *SimRank(L_1,L_2)* in Equation 3.6. The coefficient ranges from -1 to 1. A value close to 1 means a perfect positive correlation between the two rankings and a value close to -1 means a perfect negative correlation. If the two rankings have almost no correlation, the correlation coefficient will be close to zero.

(4) For each document model, we assume independence between any terms. We also assume the term frequencies in the sampled document follow Poisson distributions with the means equal to the corresponding term frequencies in the original document. Modeling term frequencies by Poisson distributions has been widely adopted by other researchers [46,47]. Furthermore, many retrieval models, such as the query likelihood model, only take query terms into account when ranking documents. In this case, we can simplify Equation 3.6 by assuming that the frequencies of non-query terms are constant in the sampled document. Formally speaking, given document $d_k = \{d_{k,1},...d_{k,i},...d_{k,V}\}$ and query $Q = \{q_1,...q_i,...q_V\}$, probability mass function f_{X_k} of document model $X_k = \{X_{k,1}, X_{k,2},...X_{k,i},..X_{k,V}\}$ is estimated as follows:

$$f_{X_k}(x_1, x_2,...x_V) = \prod_{j=1}^{V} f_{X_{k,j}}(x_j) \quad (3.8)$$

where $f_{X_{k,j}}(x)$ is given by :

$$if\,(q_j > 0)\ AND\ (D_{k,j} > 0)$$

$$f_{X_{k,j}}(x) = \Pr(X_{k,j} = x) = \frac{e^{-\lambda}\lambda^x}{x!}, x \in N, \lambda = D_{k,j}$$

$$else$$

$$f_{X_{k,j}}(x) = \Pr(X_{k,j} = x) = \begin{cases} 1, if\ \ x = D_{k,j} \\ 0, else \end{cases}$$

For better understanding, we give a toy example to show how to generate a simulated document given the original document based on the above assumptions.

The basic idea is to perturb document term counts according to the Poisson distribution.

Given vocabulary $V=\{a,b,c\}$, query $Q=\{a\}$ and document $D_1=\{a,a,b,b,b\}$, Q and D_1 are represented by 3-dimension vector [1,0,0] and [2,3,0] respectively. Let N(D_1) denotes a simulated document generated from X_1 ,that is, the document model of D_1 . Since term c does not occur in D_1 , it will not occur in N(D_1). Since term b is a non-query term and it occurs three times in D_1, it will occur exactly three times in N(D_1). The occurrence frequency of term a in N(D_1) is a random number determined by Poisson distribution P(λ) with λ=2 because term a occurs twice in D_1. For example, $\{a,a,a,b,b,b\}$ and $\{a,b,b,b\}$ are two possibilities of N(D_1).

(5) The expectation in Equation 3.6 is very hard to evaluate directly. Instead, we independently draw K samples $T(1),T(2),..T(K)$ from $f_X(T)$ to approximate the expectation, that is, Equation 3.6 is estimated as:

$$\begin{aligned}&Robustness\ Score(Q,G,C,X)\\&\cong \frac{1}{K}\sum_{i=1}^{K} SimRank(L(Q,G,C),L(Q,G,T(i))) \qquad (3.9)\end{aligned}$$

where $T(i)$ is a sample independently drawn from $f_X(T)$ which is determined by Equation 3.7 and 3.8.

The error of this estimation is proportional to the reciprocal of the square root of K [48]. According to our experiments, we find that a relatively small value of K is good and stable enough for query performance prediction.

In summary, robustness score calculation takes the following steps. First, we perform retrieval with query *Q* and retrieval function *G*. Then we generate *J* simulated documents using the document models of the top *J* documents retrieved and rank the simulated documents with the same query and retrieval function. The similarity between the two ranked lists is computed using the Spearman rank correlation coefficient. We repeat this *K* times and the average of the Spearman correlation coefficient is the robustness score.

We briefly explain why the robustness score defined above gives us useful information on retrieval performance. A low robustness score means that after document perturbation the ranking function provides a very different ranking compared to the original ranking. The low robustness score suggests that the degree of correlation between documents in the ranked list is low and the original ranking is more like a random ranking. In other words, the low robustness score is likely to correspond to a poorly- performing retrieval that returns a ranked list of loosely related topic covering many topics.

In the above discussion, we assume that the retrieval task is the ad-hoc retrieval based on topic relevance. However, with regard to named-page (NP) finding queries that usually have only a single relevant document, the expected positive correlation with query performance may not exist any more. In fact, our experiments in the next chapter will show that the use of the robustness score for performance prediction is much less effective when it comes to named-paged finding (NP) queries. This is largely due to the fact that top ranked documents in the ranked list in response to a NP query are not necessarily related while those documents are connected more or less by the topic in the case of ad-hoc retrieval.

3.2.3 First Rank Change

```
Input: (1) ranked list $L=\{D_i\}$ where $1 \le i \le 100$. $D_i$ denotes the i-th ranked document. (2) query Q
1 initialize: (1) set the number of trials J=100000 (2) counter c=0;
2 for i=1 to J
3     Perturb every document in L, let the outcome be a set $F=\{D_i'\}$ where $D_i'$ denotes the perturbed version of $D_i$.
4     Do retrieval with query Q on set F
5     c=c+1 if and only if $D_1'$ is ranked first in step 4
6 end of for
7 return the ratio c/J
```

Figure 3.4: Pseudo-code for computing FRC

To measure ranking robustness for named-page (NP) finding queries, we propose a method called the *first rank change* (FRC) which is derived from the robustness score technique. As we described above, the robustness score is not very effective for NP queries because it takes the top ranked documents as a whole into account while NP queries usually have only one single relevant document. Instead, FRC focuses on the first-ranked document, since the quality of this first-ranked document dominates retrieval effectiveness for the named-page finding task. Specifically, the pseudo-code for computing FRC is shown in figure 3.4.

FRC approximates the probability that the first-ranked document in the original list L will remain ranked first even after the documents are perturbed. The higher the probability is, the more confidence we have in the first ranked document. On the other hand, in the extreme case of a random ranking, the probability would be as low as 0.5. We expect that FRC has a positive association with NP query performance. The document perturbation step (step 3) is the same as that used in robustness score computation (we refer the reader to Equation 3.8 and the toy example given in section 3.2.2).

3.3 Query Feedback

In this section, we introduce our third prediction technique called *query feedback* (QF). Suppose that a user issues query Q to a retrieval system and a ranked list L of documents is returned. We view the retrieval system as a noisy channel. Specifically, we assume that the output of the channel is L and the input is Q. After going through the channel, Q becomes corrupted and is transformed to ranked list L.

By thinking about the retrieval process this way, the problem of predicting retrieval effectiveness turns to the task of evaluating the quality of the channel. In other words, prediction becomes finding a way to measure the degree of corruption that arises when Q is transformed to L. As directly computing the degree of the corruption is difficult, we tackle this problem by approximation. Our main idea is that we measure to what extent information on Q can be recovered from L on the assumption that only L is observed. Specifically, we design a decoder that can accurately translate L back into new query Q' and the similarity S between the original query Q and the new query Q' is adopted as a performance predictor. This is a sketch of how the QF technique predicts query performance. Before filling in more details, we briefly discuss why this method would work.

There is a relation between the similarity S defined above and retrieval performance. On the one hand, if the retrieval has strayed from the original sense of the query Q, the new query Q' extracted from ranked list L in response to Q would be very different from the original query Q. On the other hand, a query distilled from a ranked list containing many relevant documents is likely to be similar to the original query. Further examples in support of the relation will be provided later.

Next we detail how to build the decoder and how to measure the similarity S.

In essence, the goal of the decoder is to compress ranked list L into a few informative terms that should represent the content of the top ranked documents in L. Our approach to this goal is to represent ranked list L by a language model (distribution over terms). Then terms are ranked by their contribution to the language model's KL (Kullback-Leibler) divergence from the background collection model (that is, clarity contribution). Top ranked terms will be chosen to form the new query Q'. This approach is similar to that used in Section 4.1 of [49].

Specifically, we take three steps to compress ranked list L into query Q' without referring to the original query.

1. We adopt the *ranked list language model* [50], to estimate a language model based on ranked list L. The model can be written as:

$$P(w \mid L) = \sum_{D \in L} P(w \mid D) P(D \mid L) \qquad (3.10)$$

where w is any term and D is a document. $P(D \mid L)$ is estimated by a linearly decreasing function of the rank of document D.

2. Each term in P(w|L) is ranked by the following KL-divergence contribution:

$$P(w \mid L) \log \frac{P(w \mid L)}{P(w \mid C)} \qquad (3.11)$$

where P(w|C) is the collection model estimated by the relative frequency of term w in collection C as a whole .

3. The top N ranked terms by Eq.3.11 form a weighted query Q'={(w_i,t_i)} i=1,N. where w_i denotes the i-th ranked term and weight t_i is the KL-divergence contribution of w_i in Eq. 3.11.

Two representative examples, one for a poorly performing query “Cruise ship damage sea life” (TREC topic 719; average precision: 0.08) and the other for a high performing query “prostate cancer treatments”(TREC topic 710; average precision: 0.49), are shown in Table 3.1 and 3.2 respectively. These examples indicate how the similarity between the original and the new query correlates with retrieval performance. The parameter N in step 3 is set to 20 empirically and choosing a larger value of N is unnecessary since the weights after the top 20 are usually too small to make any difference.

Table 3.1 : Top 5 terms compressed from the ranked list in response to query "Cruise ship damage sea life"

Term	cruise	ship	vessel	sea	passenger
KL contribution	0.050	0.040	0.012	0.010	0.009

Table 3.2: Top 5 terms compressed from the ranked list in response to query "prostate cancer treatments"

Term	prostate	cancer	treatment	men	therapy
KL contribution	0.177	0.140	0.028	0.025	0.020

To measure the similarity between original query Q and new query Q', we first use Q' to do retrieval on the same collection. A variant of the query likelihood model [41] is adopted for retrieval. Namely, documents are ranked by:

$$P(Q'|D) = \sum_{(w_i,t_i)\in Q'} P(w_i|D)^{t_i} \qquad (3.12)$$

where w_i is a term in Q' and t_i is the associated weight. D is a document.

Let L' denote the new ranked list returned from the above retrieval. The similarity is measured by the overlap of documents in L and L'. Specifically, the percentage of documents in the top K documents of L that are also present in the top K documents in L’. the cutoff K is treated as a free parameter.

We summarize here how the QF technique predicts performance given a query Q and the associated ranked list L. We first obtain a weighted query Q’ compressed from L by the above three steps. Then we use Q’ to perform retrieval and the new ranked list is L’. The overlap of documents in L and L’ is used for prediction. In addition, QF is mainly designed for content-based queries.

3.4 Weighted Information Gain

This section introduces a weighted information gain approach that incorporates both single term and proximity features for predicting performance for both content-based and Named-Page (NP) finding queries.

Given a set of queries Q={Q_s} (s=1,2,..N) which includes all possible user queries and a set of documents D={D_t} (t=1,2…M), we assume that each query-document pair (Q_s,D_t) is manually judged and will be put in a relevance list if Q_s is found to be relevant to D_t. The joint probability P(Q_s,D_t) over queries Q and documents D denotes the probability that pair (Q_s,D_t) will be in the relevance list. Such assumptions are similar to those used in [51]. Assuming that the user issues query $Q_i \in Q$ and the retrieval results in response to Q_i is a ranked list L of documents, we calculate the amount of information contained in P(Q_s,D_t) with respect to Q_i and L by Eq.3.13

which is a variant of entropy called the weighted entropy[52]. The weights in Eq.3.13 are solely determined by Q_i and L.

$$H_{Q_i,L}(Q_s,D_t) = -\sum_{s,t} weight(Q_s,D_t)\log P(Q_s,D_t) \qquad (3.13)$$

Since we are given the user's query Q_i and the associated ranked list L, we choose the weights as follows:

$$weight(Q_s,D_t) = \begin{cases} 1/K, if \;\; s=i \; and \; D_t \in T_K(L) \\ 0, otherwise \end{cases} \qquad (3.14)$$
$$where \; T_K(L) \; contains \; the \; top \; K \; documents \; in \; L$$

The cutoff rank K is a parameter in our model.

Noticing that weight(Q_s,D_t) sums up to 1 over s and t, Eq.3.13 can be rewritten as follows:

$$H_{Q_i,L}(Q_s,D_t) = -\frac{1}{N}\sum_{s,t} W(Q_s,D_t)P(Q_s,D_t)\log P(Q_s,D_t) \qquad (3.25)$$
$$where \quad W(Q_s,D_t) = \frac{weight(Q_s,D_t)}{P(Q_s,D_t)}, N = \sum_{s,t} W(Q_s,D_t)P(Q_s,D_t)$$

When weight(Q_s,D_t) is equal to P(Q_s,D_t), weighted entropy $H_{Q_i,L}(Q_s,D_t)$ reduces to the ordinary Sharon entropy which is the expected value of log P(Q_s,D_t), a measure of the uncertainty associated with distribution P(Q_s,D_t). From Eq. 3.25, we can see weighted entropy $H_{Q_i,L}(Q_s,D_t)$ can be viewed as an expectation value of the same quantity with a weighted version of P(Q_s,D_t) being used to calculate the expectation value. That is, weighted entropy $H_{Q_i,L}(Q_s,D_t)$ measures the amount of information (uncertainty) associated with P (Q_s,D_t) with respect to the weights decided by query Q and ranked list L. Specifically, since P (Q_s,D_t) denotes the probability that top ranked document D_t in L will be relevant to query Q, weighted entropy $H_{Q_i,L}(Q_s, D_t)$ can represent the amount of information about how likely the top ranked documents in L would be relevant to query Q_i on average.

When plugging Eq. 3.14, weighted entropy $H_{Q_i,L}(Q_s, D_t)$ defined in Eq.3.13 can be simplified as follows:

$$H_{Q_i,L}(Q_s, D_t) = \frac{1}{K} \sum_{D_t \in T_K(L)} -\log P(Q_i, D_t) \qquad (3.15)$$

If we view the term -log $P(Q_i,D_t)$ in Eq. 3.15 as the relevance score of document D_t with respect to query Q_i , another interpretation of weighted entropy $H_{Q_i,L}(Q_s, D_t)$ is the average relevance score over the top K ranked documents for a given query. Therefore, weighted entropy $H_{Q_i,L}(Q_s, D_t)$ can also represent the retrieval quality of the given ranked list L. However, we want to point out that the average relevance score cannot be interoperated as weighted entropy if the score is not in the form of a logarithm of a probability (if it is, for example, a tf-idf score).

Unfortunately, weighted entropy $H_{Q_i,L}(Q_s, D_t)$ computed by Eq.3.15, cannot be compared across different queries, making it inappropriate for directly predicting query performance. To mitigate this problem, we come up with a background distribution $P(Q_s,C)$ over Q and D by imagining that every document in D is replaced by the same special document C which represents average language usage. In this thesis, C is created by concatenating every document in D. Roughly speaking, C is the collection (the document set) $\{D_t\}$ without document boundaries. Similarly, weighted entropy $H_{Q_i,L}(Q_s, C)$ calculated by Eq.3.15 represents the amount of information about how likely an average document (represented by the whole collection) would be relevant to query Q_i. Since a "ranked list" of this average document can be viewed as a random ranking of documents in the collection on average, $H_{Q_i,L}(Q_s, C)$ can be thought of representing the average retrieval quality of a random ranked list of documents.

Now we introduce our performance predictor *WIG* which is the weighted information gain [52] computed as the difference between $H_{Q_i,L}(Q_s, D_t)$ and $H_{Q_i,L}(Q_s, C)$. Specifically, given query Q_i, collection C and ranked list L of documents, WIG is calculated as follows:

$$\begin{aligned} WIG(Q_i, C, L) &= H_{Q_i,L}(Q_s, C) - H_{Q_i,L}(Q_s, D_t) \\ &= \sum_{s,t} weight(Q_s, D_t) \log \frac{P(Q_s, D_t)}{P(Q_s, C)} = \frac{1}{K} \sum_{D_t \in T_K(L)} \log \frac{P(Q_i, D_t)}{P(Q_i, C)} \end{aligned} \quad (3.16)$$

From the viewpoint of information theory, with respect to a given query and its corresponding ranked list, WIG computed by Eq.3.16 measures the change in information (uncertainty) associated with the probability P(Q, D) (the probability that query Q is relevant to document D) from an imaginary state that all documents are replaced by a special "average" document to a posterior state that the actual documents are restored. On the other hand, from the viewpoint of IR (Information retrieval), WIG can be viewed as the difference of the average document score between the actual ranked list and a random ranked list. We hypothesize that WIG is positively correlated with retrieval effectiveness because high quality retrieval should be much more effective than just randomly ranking the documents in the collection.

The heart of this technique is how to estimate the joint distribution P(Q_s,D_t). In the language modeling approach to IR, a variety of models can be applied readily to estimate this distribution. Although most of these models are based on the bag-of-words assumption, recent work on modeling term dependence under the language modeling framework have shown consistent and significant improvements in retrieval effectiveness over bag-of-words models. Inspired by the success of incorporating term proximity features into language models, we decide to adopt a good dependence model to estimate the probability P(Q_s,D_t). The model we chose for this paper is Metzler and Croft's Markov Random Field (MRF) model, which has already demonstrated superiority over a number of collections and different retrieval tasks [51,53].

According to the MRF model, log P(Q_i, D_t) can be written as

$$\log P(Q_i, D_t) = -\log Z_1 + \sum_{\xi \in F(Q_i)} \lambda_\xi \log P(\xi \mid D_t) \qquad (3.17)$$

where Z_1 is a constant that ensures that P(Q_i, D_t) sums up to 1. F(Q_i) consists of a set of features expanded from the original query Q_i . For example, assuming that query Q_i is "talented student program", F(Q_i) includes features like "program" and "talented student". We consider two kinds of features: single term features T and proximity features P. Proximity features include exact phrase (#1) and unordered window (#uwN) features as described in [51]. Note that F(Q_i) is the union of T(Q_i) and P(Q_i). For more details on F(Q_i) such as how to expand the original query Q_i to F(Q_i), we refer the reader to [51] and [53]. P(ξ|D_t) denotes the probability that feature ξ will occur in D_t. More details on $P(\xi|D_t)$ will be provided later in this section. The choice of λ_ξ is somewhat different from that used in [51] since λ_ξ plays a dual role in our model. The first role, which is the same as in [51], is to weight between single term and proximity features. The other role, which is specific to our prediction task, is to normalize the size of $F(Q_i)$.We found that the following weight strategy for λ_ξ satisfies the above two roles and generalizes well on a variety of collections and query types.

$$\lambda_\xi = \begin{cases} \dfrac{\lambda_T}{\sqrt{|T(Q_i)|}}, \xi \in T(Q_i) \\ \dfrac{1-\lambda_T}{\sqrt{|P(Q_i)|}}, \xi \in P(Q_i) \end{cases} \qquad (3.18)$$

where $|T(Q_i)|$ and $|P(Q_i)|$ denote the number of single term and proximity features in $F(Q_i)$ respectively. The reason for choosing the square root function in the denominator of λ_ξ is to penalize a feature set of large size appropriately, making WIG more comparable across queries of various lengths. λ_T is a fixed parameter and set to 0.8 according to [51] throughout this thesis.

Similarly, log $P(Q_i,C)$ can be written as:

$$\log P(Q_i, C) = -\log Z_2 + \sum_{\xi \in F(Q_i)} \lambda_\xi \log P(\xi | C) \qquad (3.19)$$

When constant Z_1 and Z_2 are dropped, WIG computed in Eq.3.16 can be rewritten as follows by plugging in Eq.3.18 and Eq.3.19:

$$WIG(Q_i, C, L) = \frac{1}{K} \sum_{D_t \in T_K(L)} \sum_{\xi \in F(Q_i)} \lambda_\xi \log \frac{P(\xi | D_t)}{P(\xi | C)} \qquad (3.20)$$

One of the advantages of WIG over other techniques is that it can handle well both content-based and NP queries. Based on the type (or the predicted type) of Q_i, the calculation of WIG in Eq. 3.20 differs in two aspects: (1) how to estimate $P(\xi|D_t)$ and $P(\xi|C)$, and (2) how to choose K.

For content-based queries, $P(\xi|C)$ is estimated by the relative frequency of feature ξ in collection C as a whole. The estimation of $P(\xi|D_t)$ is the same as in [51]. Namely, we estimate $P(\xi|D_t)$ by the relative frequency of feature ξ in D_t linearly smoothed with collection frequency $P(\xi|C)$. K in Eq.3.20 is treated as a free parameter. Note that K is the only free parameter in the computation of WIG for content-based queries because all parameters involved in $P(\xi|D_t)$ are assumed to be fixed by taking the suggested values in [51].

Regarding NP queries, we make use of document structure to estimate $P(\xi|D_t)$ and $P(\xi|C)$ by the so-called *mixture of language models* proposed in [54] and incorporated into the MRF model for Named-Page finding retrieval in [53]. The basic idea is that a document (collection) is divided into several fields such as the title field, the main-body field and the heading field. $P(\xi|D_t)$ and $P(\xi|C)$ are estimated by a linear combination of the language models from each field. We refer the reader to [53] for details. We adopt the exact same set of parameters as used in [53] for estimation. With regard to K in Eq.3.20, we set K to 1 because the named-page finding task heavily focuses on the first ranked document. Consequently, there are no free parameters in the computation of WIG for NP queries.

3.5 Summary of Performance Prediction Techniques

In the previous four sections we introduced several techniques for performance prediction. Here we summarize and compare these methods in Table 3.3. For better comparison, we also include the JSD method [8] which was proposed by other researchers and will be used as one of our baselines later.

Table 3.3 : Comparison of Prediction Techniques
***CB*: Content-Based queries, *NP*: Named-Page finding queries, *QM*: Query Language Model, *CM*: Collection Language Model, *RS*: Robustness Score, *FRC*: First Rank Change, *WIG*: Weighted Information Gain.**

Technique	Key ideas	Designed for
Clarity	KL-divergence between QM and CM	CB
JSD	Jensen-Shannon Divergence between QM [6] and CM	CB
Ranking Robustness (RS and FRC)	Perturb terms in the top ranked documents	RS: CB FRC:NP
Query Feedback	Similarity between the original query and the new query based on clarity contribution	CB
WIG	The difference between two weighted entropies	CB and NP

The clarity technique measures ranked list coherence by the KL-divergence between the query language model based on the top retrieved documents (in response

[6] The JSD method differs from the clarity technique in how QM is estimated.

to a given query) and the collection language model representing general language usage. Similar to clarity, the JSD method uses the Jensen-Sharon Divergence (JSD) to compute the distance between the top ranked documents and the collection. The ranking robustness technique (robustness score and first rank change) measures the sensitivity of the ranking to the perturbing of the top ranked documents. Query feedback compares the similarity between the original query and the new query distilled from the ranked list. WIG (weighted information gain) measures the difference between weighted entropy H_1 based on the actual ranked list and weighted entropy H_2 based on a random ranked list (simulated by treating the whole collection as a single document).

Since both clarity and WIG measure the distance between an object representing the retrieved documents and an object representing the whole collection, we would like to explore the relationship between the two techniques in depth.

When we substitute Eq. 3.5 by Eq 3.1, the clarity score can be rewritten as follows[7]:

$$clarity\,score = \sum_{w \in V} \sum_{D \in L} P(D|Q)P(w|D) \log \frac{\sum_{D \in L} P(D|Q)P(w|D)}{P_{coll}(w)} \qquad (3.25)$$

As a comparison, we rewrite the calculation of WIG which is given in Eq. 3.20:

$$WIG(Q_i, C, L) = \frac{1}{K} \sum_{D_t \in T_K(L)} \sum_{\xi \in F(Q_i)} \lambda_\xi \log \frac{P(\xi|D_t)}{P(\xi|C)} \qquad (3.20)$$

[7] L represents the ranked list obtained by query likelihood retrieval

We can see Eq. 3.25 is very similar to Eq. 3.25. In fact, given query Q , collection C and ranked list L, Eq 3.25 and Eq. 3.20 can be written in the same form as follows:

$$score(Q,L,C) = \sum_{\xi \in T} \sum_{D \in L} weight(\xi, D) \log \frac{P(\xi, D)}{P_{coll}(\xi)}, \quad (3.26)$$

where T is a feature space , ξ is a feature in T, D is a document in L, $P(\xi, D)$ is a distribution over feature space T and $P_{coll}(\xi)$ denotes the probability that feature ξ will occur in collection C.

Clarity and WIG differ in the following three aspects:

(1) The feature space T

For clarity, the feature space is the whole vocabulary consisting of single terms. For WIG, the feature space is single terms or phrases that extracted from query Q.

(2) The $weight(\xi, D)$

For clarity score, $weight(\xi, D) = P(D \mid Q) P(\xi \mid D)$.

For WIG, $weight(\xi, D) = \begin{cases} \dfrac{\lambda_\xi}{K}, if\ D\ is\ one\ of\ the\ top\ K\ documents\ in\ L \\ 0, otherwise \end{cases}$.

We can see that the $weight(\xi, D)$ for WIG is a almost a constant and is much simpler than that for clarity, which makes WIG free from estimation noise in $P(\xi \mid D)$ and $P(D \mid Q)$.

(3) $P(\xi, D)$

For the clarity score, $P(\xi, D) = \sum_{D \in L} P(D \mid Q) P(\xi \mid D)$.

For WIG, $P(\xi, D) = P(\xi \mid D)$.

This means that the clarity score uses a document model averaged over all documents in the ranked list for $P(\xi, D)$, while WIG use the actual document model of document D.

3.6 Model Comparison for Query Expansion Prediction

Query expansion is an effective technique for improving average IR performance. However, one problem with this approach is that it can greatly hurt the performance of some individual queries when many unrelated terms are added during expansion. Motivated by this issue, in this section we address the problem of detecting the case when using the results of an expansion technique hurts the retrieval performance for that particular query. In other words, we would like to predict the performance change (either positive or negative change) between the two retrieval strategies (that is, the unexpanded and the expended retrieval) for a given query. We call this task *query expansion prediction*. Based on this prediction, we can selectively apply the query expansion technique on a per-query basis.

In the previous sections, we have discussed our models for query performance prediction. One major difference between query performance prediction and query expansion prediction is as follows: In the former, the retrieval function is usually assumed to be fixed and we compare retrieval effectiveness across queries. In the latter, essentially we compare the retrieval effectiveness between two retrieval functions for a given particular query. Simply applying techniques for query performance prediction is not sufficient for query expansion prediction. The main reason is that the performance of a query is not directly related to whether or not we should apply expansion for that query. For example, one might expect that query expansion would improve poorly-performing queries. However, this is not always the case according to our observation on many TREC test collections. Query expansion is based on the assumption that the top retrieved documents are relevant. However, when the initial retrieval is not of high quality (that is, includes many irrelevance documents), it is unclear whether or not terms extracted from the initial results will be related to the original query. We also tried the difference between the predicted performance of the unexpanded retrieval (by some of our prediction techniques) and that of the expanded retrieval for query expansion prediction, but this strategy does not work well. Instead of applying a performance prediction technique, we propose a method called *model comparison* inspired by clarity score-related ideas for query expansion prediction.

For the query expansion prediction task we are interested in predicting which of the two ranked lists yield better performance for a given query: the unexpanded retrieval rank list or the expanded one. For each ranked list, we estimate a ranked list language model to represent the language usage in the ranked list as we did in our query feedback technique (Eq. 3.10). We then compare the two ranked list models. With this comparison, our goal is to catch when the expanded retrieval has deviated from the original sense of the query by calculating the distance between the two language models of the unexpanded and expanded results.

Specifically, given query Q, we assume the unexpanded retrieval results are presented by a ranked document list A. Similarly, the expanded retrieval results are presented by a ranked document list B. We build two ranked list language models for these two ranked list respectively. That is, applying the ranked list language model defined in Eq. 3.10 to ranked list A and B, we have

$$P(w|A) = \sum_{D \in A} P(w|D)P(D|A) \qquad (3.21)$$

$$P(w|B) = \sum_{D \in B} P(w|D)P(D|B) \qquad (3.22)$$

To estimate document model $P(w|D)$, we use Equation 3.2 with Dirichlet smoothing.

The model comparison score is calculated as follows:

$$comparison\ score = \sum_{w \in \tau} P(w|A) \log_2 \frac{P(w|A)}{P(w|B)} \qquad (3.23)$$

where τ represents the set of top T terms in contribution to the clarity score of Model A. That is, given query Q, we compute the score of each term as

$$contrib(w|Q) = P(w|Q) \log_2 \frac{P(w|Q)}{P(w|coll)} \qquad (3.24)$$

and take the T highest terms. These terms are considered to be important terms in terms of describing the query. The choice of using the top T terms rather than all vocabulary terms is due to the observation that comparing the two models on generic terms can bring noise to the comparison score.

The model comparison score defined in Eq. 3.24 is an expectation of the difference in log probabilities for the important unexpanded terms in models of the two ranked lists. We briefly discuss why the comparison score can be helpful for query expansion prediction. When the important terms are used much less frequently in the expanded model, the comparison score will be positively high. This often indicates an expansion that has strayed from the original sense of the query. On the other hand, a negative score indicates that the expanded retrieval uses the important terms more frequently, which often indicates that the expanded result is better than the other.

CHAPTER 4

EVALUATION OF PREDICTION MODELS

In this chapter, we thoroughly evaluate our query performance prediction techniques across a variety of search settings. We first report our experiments on a number of traditional TREC test collections that focus on the ad-hoc retrieval task. Then we consider web search environments that are significantly different from the traditional TREC settings in many aspects such as query types. In addition, we show results of the model comparison technique for query expansion prediction.

4.1 Query Performance Prediction in Traditional TREC Settings

In this thesis, “traditional TREC settings” refer to the following: (1) ad-hoc retrieval which is the task of finding a number of documents that are relevant to a particular information need, and (2) TREC document collections which typically consist of no more than one million relatively homogenous newswire articles. Most previous work on query performance prediction has focused on these traditional TREC settings.

4.1.1 Experimental Setup

Our experiments use a variety of TREC collections. The retrieval task is the ad-hoc retrieval. Queries used in our experiments are titles of TREC topics unless explicitly noted. Table 4.1 gives the summary of these test collections.

Table 4.1: Summary of test collections used in Section 4.1

TREC	Collection	Topic Number	Number of Document
1+2+3	Disk 1+2+3	51-150	1,078,166
4	Disk 2+3	201-250	567,529
5	Disk 2+4	251-300	524,929
Robust 2004	Disk 4+5 minus CR	301-450; 601-700[8]	528,155

In this section, we evaluate four of our models, that is, clarity score, robustness score, QF (query feedback) and WIG (weighted information gain).

For computing the clarity score, the document model P(w|D) in Equation 3.2 is estimated by using Dirichlet smoothing with Dirichlet prior μ=1000. Document models are mixed from Jelinek-Mercer smoothed document models with λ=0.6.

With regard to robustness score calculation, we use the query likelihood model [41] with Dirichlet smoothing as the ranking function (Dirichlet prior μ is again set to 1000). We set parameter K in Equation 3.9 to 100. We tried different values of K ranging from 10 to 500000 and found that the results change very little starting from 100. This means we do not have to require a large number of samples to compute robustness scores. In addition, for all of the test collections we choose the top 50 documents to compute the rank similarity in Equation 3.9. We observe that the prediction performance using the robustness score is relatively insensitive to the choice of the cutoff rank as long as it is in the range of 30 to 100.

[8] Topic 672 is removed because of no relevant documents.

Regarding Query Feedback (QF) calculation, we empirically set the parameter N (that is, the top N terms ranked by Equation 3.11) to 20. In fact, choosing a larger value of N is unnecessary since the weights after the top 20 are usually too small to make any difference. When comparing the overlap between the old and the new ranked list, we need to specify the cutoff rank K. We set K to 25 for all of our test collections. Further investigation shows that any value of K ranging from 20 to 50 will lead to satisfactory prediction accuracy in most cases.

Regarding WIG computation, there is only one free parameter, that is, the cutoff rank in Equation 3.20. We find that a small value of K (a value less than 10) gives good prediction in most cases and accordingly we set K to 5 for all of the collections.

To evaluate the accuracy of query performance prediction, we measure the correlation between the predicted and the actual retrieval performance for a set of test queries, which is widely adopted in most prediction work. In essence, we predict the relative performance of retrieval instead of the actual performance, considering the fact that prediction accuracy may be affected by the choice of performance measure if we choose to predict the actual retrieval performance.

Next we provide more details on the evaluation of prediction power. Given query set $Q = \{Q_1, Q_2, ...Q_n\}$ containing n queries and a document collection, we perform retrieval on the collection and have $X = \{X_1, X_2, ...X_n\}$ where X_i denotes the actual retrieval performance of query Q_i. Since our retrieval task is ad-hoc retrieval in this section, we choose average precision for measuring performance. That is, X_i is the average precision of Q_i. For a given prediction technique to be evaluated, we then compute $Y = \{Y_1, Y_2, ...Y_n\}$ where Y_i is the score computed by the prediction technique for query Q_i. Our goal is to measure the extent of dependence between X and Y. The higher the dependence is, the more accuracy the prediction will be.

To this end, we use both the Pearson's ρ correlation test [55] and the Kendall's τ rank correlation test[45]. Pearson's correlation reflects the degree of linear relationship between the two variables X and Y. The formula for computing Pearson's correlation coefficient ρ is as follows:

$$\rho = \frac{\sum XY - \frac{\sum X \sum Y}{n}}{\sqrt{(\sum X^2 - \frac{(\sum X)^2}{n})(\sum Y^2 - \frac{(\sum Y)^2}{n})}} \qquad (4.1)$$

Let $R(X)$ denote the ranking of X, that is, $R(X) = \{R_{X_1}, R_{X_2}, ..R_{X_n}\}$ where R_{X_i} is the rank of X_i in X. Similarly, let $R(Y)$ denote the ranking of Y. Instead of directly measuring the correlation between X and Y, Kendall's τ rank correlation tests the agreement between $R(X)$ and $R(Y)$ to evaluate the magnitude of the correlation. Kendall's rank correlation test is a non-parametric test since it does not assume any distributions of both variables. Kendall τ coefficient is computed by:

$$\tau = \frac{C - D}{\frac{1}{2}n(n-1)} \qquad (4.2)$$

where C denotes the number of concordance (correctly-ordered) pairs and D denotes the number of discordance (incorrectly-ordered) pairs. The total number of pairs is $\frac{1}{2}n(n-1)$

The values of both kinds of correlation range between -1.0 and 1.0 where -1.0 means perfect negative correlation and 1.0 means perfect positive correlation.

Although the Kendall’s test and the Pearson’s test are widely adopted in the evaluations of many prediction techniques, there are other ways to measure the correlations between two variables. One example is the Spearman’s rank correlation test [45]. Our experience is that it does not matter which correlation measure to use, since these measures generally give similar results.

To obtain average precision, document retrieval is done by using the query-likelihood model [41] and the results are evaluated by the trec_eval program. Again, Dirichlet smoothing with Dirichlet prior μ =1000 is used for smoothing.

4.1.2 Experimental Results

The results for correlation with average precision measured by the Pearson and Kendall test are presented in Table 4.2 and 4.3 respectively.

From these results, for all of our four prediction models we observe statistically significant correlation with average precision over all of the test collections no matter which correlation coefficient is adopted. To better understand how these numbers translate to the strength of correlation, we plot average precision versus robustness scores on Robust04 in Figure 4.1. We clearly observe a linear trend between the predicted and the actual retrieval performance.

Table 4.2: Pearson's correlation coefficient for correlation with average precision, for clarity score(Clarity), robustness score (Robustness), Weighted Information Gain (WIG) and Query Feedback(QF). Bold cases mean the results are statistically significant at the 0.05 level. Stars represent the strongest correlation for that collection.

TREC	Clarity	Robustness	WIG	QF
TREC123	**0.335**	**0.434**	**0.494***	**0.484**
TREC5	**0.366**	**0.454**	**0.432**	**0.530***
Robust 04	**0.507**	**0.550***	**0.456**	**0.499**

Table 4.3 : Kendall's correlation coefficient for correlation with average precision, for clarity score(Clarity), robustness score(Robustness), Weighted Information Gain (WIG) and Query Feedback(QF). Bold cases mean the results are statistically significant at the 0.05 level. Stars represent the strongest correlation for that collection.

TREC	Clarity	Robustness	WIG	QF
TREC123	**0.331**	**0.329**	**0.407***	**0.339**
TREC5	**0.311**	**0.328**	**0.317**	**0.396***
Robust 04	**0.412***	**0.392**	**0.370**	**0.355**

We also notice that there is no single predictor that always performs the best or the worst consistently on all of the three collections. This suggests that the prediction power of our models is roughly at the same level on these data sets.

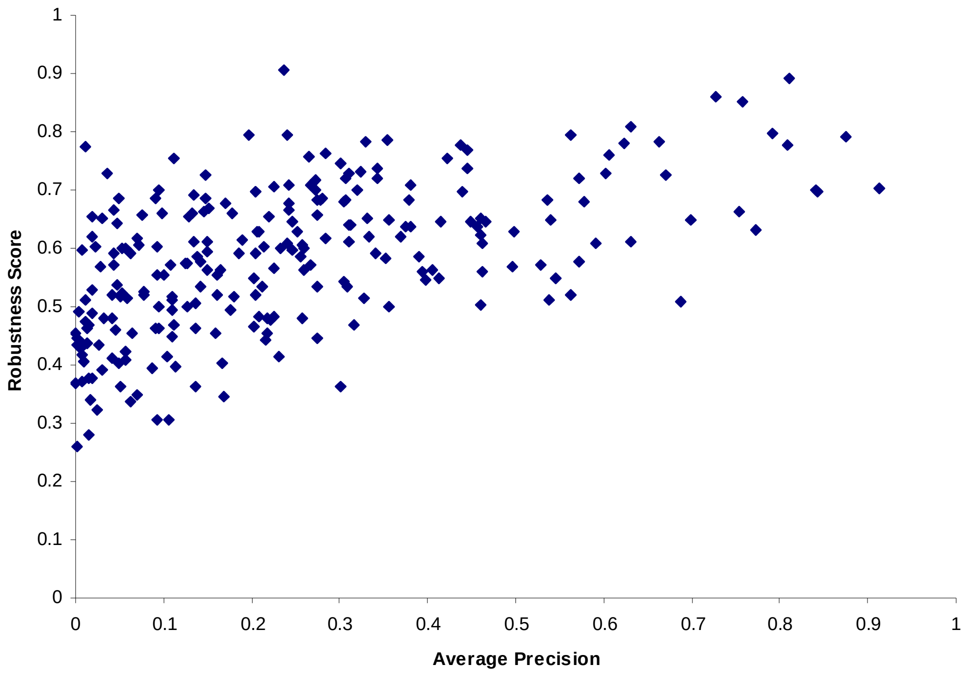


Figure 4.5 : **Average precision versus robustness scores for the 249 title queries from the Robust 2004 Track**

When comparing Table 4.2 to Table 4.3, we can see that results obtained by using the Pearson's test are highly consistent with those obtained by using the Kendall's test. For example, the two tests always agree on which one is the best predictor for a given test collection. Therefore, it is not necessary to include both measures for measuring prediction accuracy. In addition, similar to Pearson's correlation measuring linear correlation between two variables, R-square (or coefficient of determination) measures how well a linear model fits data. Although not reported here, we also tried R-square in some of our experiments and found that the results are consistent to those obtained by Pearson's correlation. For more details, we refer the reader to section 4.3 of our paper [6].

All queries used in the above experiments are the title part of TREC topics that typically consists of two or three key words. In this thesis we focus on this kind of query, since short queries dominate current search engines. However, it would be interesting to see how our techniques will perform when it comes to long queries that typically consist of one or more natural English sentences. To this end, we evaluate our techniques on two data sets: TREC4 and Robust 2004. We use the description part of the topics as our test queries. To see how description queries may be different from title queries, we take TREC topic 301 for example. The title part of this topic is as follows: *International Organized Crime*. The description part is as follows: *Identify organizations that participate in international criminal activity, the activity, and, if possible, collaborating organizations*. Other than queries, all settings (such as prediction model parameters, retrieval parameters) are the same as those used for title queries in our previous experiments. Here we choose the Kendall's τ for evaluating prediction accuracy and the results are presented in Table 4.4.

Again, all of our predictors show significant correlation with retrieval performance. However, from Table 4.4 we observe that WIG does not perform well compared to the other three models. To better understand how query types can have an impact on our prediction techniques, Figure 4.2 compares Kendall coefficients for title queries to those for description queries on the Robustness 2004 Track.

Table 4.4 : Kendall's correlation coefficient for correlation with average precision, for clarity score(Clarity), robustness score(Robustness), Weighted Information Gain (WIG) and Query Feedback(QF). Bold cases mean the results are statistically significant at the 0.05 level. Stars represent the strongest correlation for that collection. Queries are the *description* part of TREC topics.

Collection	Clarity	Robust	WIG	QF
TREC4	**0.353**	**0.548***	**0.304**	**0.533**
Robust 04	**0.373**	**0.464***	**0.216**	**0.381**

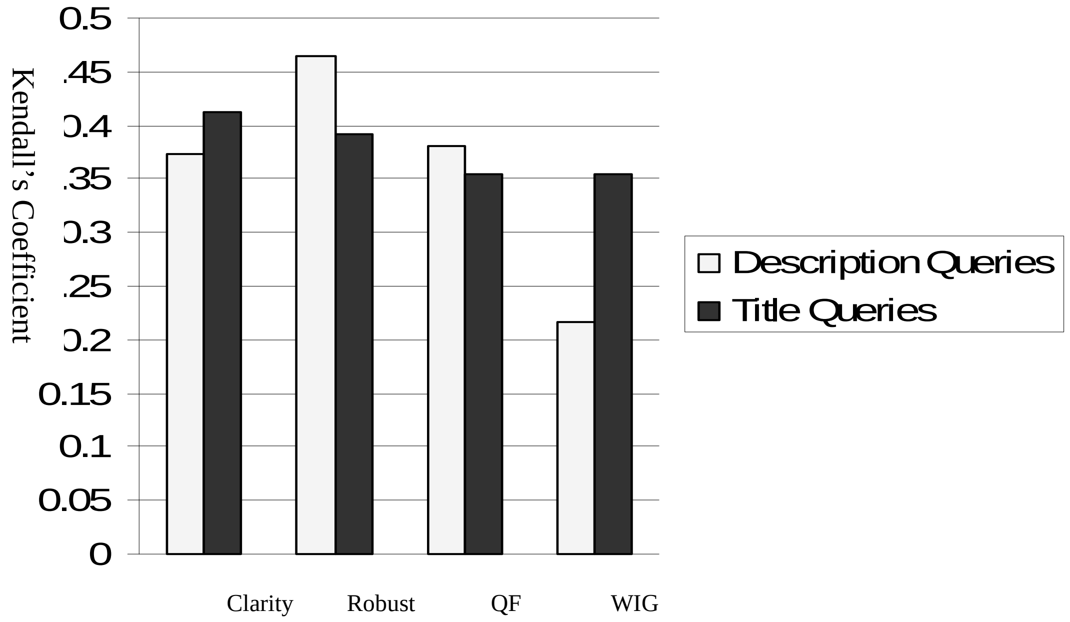


Figure 4.2 Comparison of prediction accuracy between title queries and description queries

For the first three methods, (that is, clarity score, robustness score and query feedback), they perform consistently on both types of queries, which suggests that these three models are robust to query length. Unfortunately, the performance of WIG drops noticeably when it comes to description queries.

We briefly discuss the reason that WIG is sensitive to long queries. Given a query, WIG extracts a set of features from the query (the feature can be either a term or a phrase in the query) and compare the frequencies of these features in the top ranked documents to those in the whole collection. The quality of these features has a direct impact on the prediction power of WIG. Features extracted from a title query are usually closely related to the information need (topic). This means these extracted features generally are of high quality. However, it is not the case for description queries, since the description field of a topic uses natural language to describe the topic. In fact, features extracted from non-keywords in a description query can be misleading and are likely to degrade the performance of WIG. One possible way to overcome this problem is to automatically generate a few keywords from a description query. Though long queries can be an issue for WIG, we will see in the next section that WIG is far better than other techniques in Web search environments where short queries predominate.

We compare our methods to predictor techniques proposed by other researchers. We focus on two techniques from recent SIGIR conferences. The first one (we call it the overlap method) is a technique that is based on the overlap between the top retrieved documents of the full query and those of its sub-queries (a query consists of one query-term) [6].Yom-Tov et al. [6] demonstrated that this method has prediction accuracy superior to several other predictors proposed in the Robust Track 2004. According to the learning techniques used, there are two variants of this method: tree-based and histogram-based . We choose the best settings reported in their paper. The second one is the document perturbation (DP) method proposed in [29]. In this method, a document is randomly selected from the retrieved document set. This document is perturbed by adding some amount of noise. Then, retrieval is performed using the perturbed document as a pseudo-query. They investigate how the amount of noise added will affect the ranking. The idea behind this method is somewhat similar to our ranking robustness technique. They claim to have better predication accuracy than the overlap method on Robust 2004, a popular dataset that has been used in the evaluation of many prediction techniques. We do not include the JSD method [8] as one of our baselines in this section, since their evaluation of this method is performed on a web collection. (we will use this method as our baseline in the next section that focuses on a Web search environment.)

We also chose to use Robust 2004. By doing so, we can make fair comparison using the same dataset. Both title and description queries are considered. We choose the robustness score as the basis for this comparison. The Kendall's correlation coefficients of these methods for both of the title and the description field of 249 topics used in the Robust Track 2004 are presented in Table 4.5.

Regarding the overlap method, we assume the best Kendall scores reported in their paper. Originally, they divided 249 topics into two parts: 200 old topics and 49 new topics. For each query type, they reported two Kendall's scores: one score for the first part by using four-fold cross-validation and the other score for the second part by using the first part (the 200 old topics) as training data. This is equivalent to performing five-fold cross-validation on all 249 topics. Here we combine the two Kendall's scores into one score for our comparison on all 249 topics. For the document perturbation method, we quote their results for description queries. They do not report results for title queries and the result for title queries comes from our own implementation.

Table 4.5 : Comparison to other techniques. The reported numbers are Kendall's correlation coefficients for correlation with average precision on the Robust 2004 Track for robustness score, the overlap method [6] and the document perturbation method [29].

	Robustness	Overlap	Document Perturbation
Title	0.392	0.284	0.181
Description	0.464	0.465	0.520

From Table 4.5, we can see that our predictor consistently performs well for both title queries and description queries while the two baselines suffer from either one type or the other. Moreover, our predictor works at least at the same level with or better than the best baseline in each query type.

One interesting thing about these three methods in Table 4.5 is that they are based on the same general idea: utilize the sensitivity of retrieval system to some noise intentionally introduced in the retrieval process to estimate retrieval performance. The overlap method can be seen as a way of measuring the robustness of search results to query perturbation. The document perturbation method is essentially another way of implementing query perturbation, considering the fact that both the original document randomly selected from search results and the corresponding perturbed document are used as a pseudo-query for retrieval. On the other hand, the robustness score can be seen as a kind of collection perturbation, since documents in the collection are perturbed while the query remains intact. Therefore, the results in Table 4.5 can be interpreted as the evidence that collection perturbation is more appropriate than query perturbation for the task of performance prediction.

4.2 Query Performance Prediction in Web Search Environments

Web search environments are remarkably different from the traditional TREC settings adopted in the previous section in many ways. Major differences include: (1) a Web collection is usually much larger and more heterogeneous than a traditional TREC collection, and (2) Web queries often consist of more than one type. For example, both content-based queries and named-page finding queries are popular in web retrieval.

The major goal in this section is to evaluate the prediction power of our models in a variety of Web search settings. Specifically, we consider the following cases: (1)

content-based (CB) queries, (2) named-page (NP) finding queries, and (3) the situation where the actual query type is unknown, that is, the user query can be either CB or NP. Other than evaluations on laboratory test collections with pre-defined query sets and relevance judgments, we carry out experiments on realistic Web data collected from a commercial search engine to explore the potential of our techniques to predict user preference for search results.

4.2.1 Content-based Queries

Performance prediction for content-based queries was discussed within the context of traditional TREC settings in Section 4.1. In this section our retrieval task is the same (that is, ad-hoc retrieval). But we adopt a large Web collection called "GOV2" used in the Terabyte Tracks [56]. This collection, containing about 25 million documents crawled from Web sites in the .gov domain during the year of 2004 [57], is significantly larger than the collections used previously. Other than its size, the GOV2 collection includes Web documents with a variety of styles ranging from Web pages containing only tables to well-written articles published in newspapers.

Table 4.6 : Summary of data sets for content-based queries

Name	Collection	Topic Number	Query Type
TB04-adhoc	GOV2	701-750	CB
TB05-adhoc	GOV2	751-800	CB
TB06-adhoc	GOV2	801-850	CB

In this section, we use the ad-hoc topics of the Terabyte Tracks of 2004, 2005 and 2006 and name them TB04-adhoc, TB05-adhoc and TB06-adhoc respectively. All queries used in our experiments are titles of TREC topics as most Web queries are short. Table 4.6 summarizes the above data sets.

Retrieval performance of content-based queries is measured by average precision. We make use of the Markov Random field model for ad-hoc retrieval. This model is particularly effective in Web search environments [51]. We adopt the same setting of retrieval parameters used in [51,53]. Though not reported here, we also tried the query likelihood model for ad-hoc retrieval and found that the results change little because of the very high correlation between the query performances obtained by the two retrieval models (0.96 measured by Pearson's coefficient). We evaluate prediction accuracy by the correlation between the predicted and the actual performance, which is the same as what we did in Section 4.1. To be comparable to the results reported in other researchers' papers, we adopt the Pearson's correlation test (one of the two correlation tests used in Section 4.1) to measure the correlation. As we stated in section 4.1, it does not matter much which test to choose.

Table 4.7 shows the correlation with average precision on two data sets: one is a combination of TB04-adhoc and TB05-adhoc(100 topics in total) and the other is TB06-adhoc (50 topics). Again, our models are the clarity score (clarity), the robustness score (robust), query feedback (QF) and weighted information gain (WIG). Our baseline is the JSD method (JSD) proposed by David Carmel et al.[8]

According to the results reported in their paper, this method is significantly better than the overlap method [6] used as one of our two baselines in the previous section. For the other baseline used previously (that is, the document perturbation method), we do not use it as a baseline here, since this method performs poorly for short queries. For the clarity and robustness score, we have tried different parameter settings and report the highest correlation coefficients we have found. We directly cite the result of the JSD-based method reported in [8]. Regarding WIG, we adopt the same parameters as used in Section 4.1. In fact, we find that this set of parameter settings for WIG can achieve nearly-optimal prediction accuracy in two considerably different situations. With regard to QF, we set the cutoff K to 100, a value that is significantly different from the value used in Section 4.1. Further investigation shows that the choice of K is related to collection size. For a large web collection like the GOV2, QF prefers a larger value of K such as 100, while a smaller value of K such as 25 is appropriate on traditional TREC collections that are much smaller than a Web collection.

Even though we chose the best parameters for clarity and robustness score, the prediction accuracy of the two is low compared to WIG and QF. This suggests that clarity and robustness have difficulty in adapting to a Web collection. Similar to the clarity score method, the JSD method uses the distance between the query model and the collection model to predict performance, although the two methods differ in some details such as how to calculate the distance. Therefore, it is reasonable that the prediction power of the two is on the same level as shown in Table 4.7. In short, WIG and QF are considerably more accurate than the others. As an example, Figure 4.3 depicts the strength of the correlation with average precision for WIG on the first data set. It shows a clear linear relationship between the predictor and the actual retrieval performance.

Table 4.7 : Pearson's correlation coefficients for correlation with average precision on the Terabyte Tracks (ad-hoc) for clarity score, robustness score, the JSD-based method(we directly cites the score reported in [8]), query feedback(QF) and WIG. Bold cases mean the results are statistically significant at the 0.05 level.

Methods	Clarity	Robust	QF	WIG	JSD
TB04+05 Adhoc	**0.333**	**0.317**	**0.480**	**0.556**	**0.362**
TB06 Adhoc	0.076	**0.294**	**0.422**	**0.464**	N/A

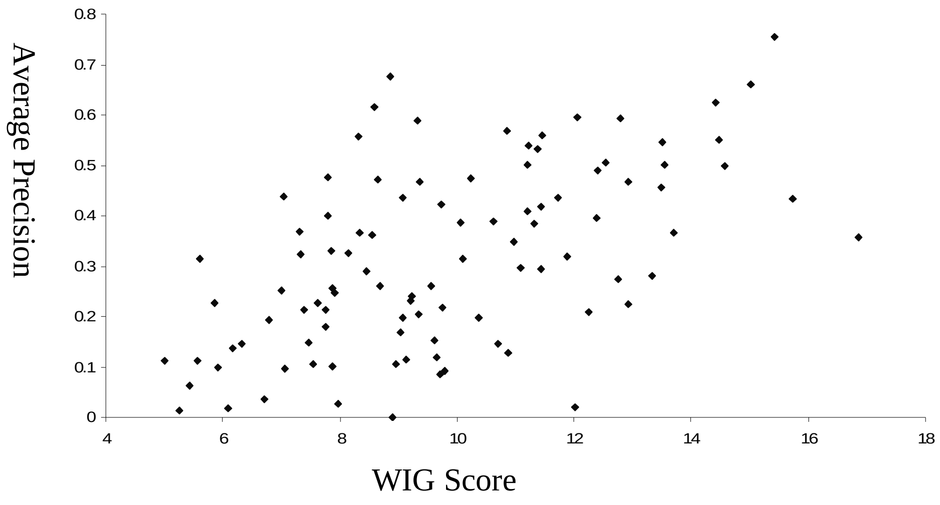


Figure 4.3: WIG score versus average precision for the 100 title ad-hoc queries from the Terabyte Tracks of 2004 and 2005

For the clarity score, we observe that it performs very poorly on TB06-adhoc. Further investigation shows that the mean average precision of TB06-ad-hoc is 0.342 and is about 10% better than that of the first data set. While the other methods typically consider the top 100 or less documents given a ranked list, the clarity method usually needs the top 500 or more documents to adequately measure the coherence of a ranked list. Higher mean average precision makes ranked lists retrieved by different queries more similar in terms of coherence at the level of top 500 documents. We believe that this is the main reason for the low accuracy of the clarity score on this data set.

To see how well our prediction models scale to large Web collections, we suggest the reader compare Table 4.7 to Table 4.3. One the one hand, we are glad to find that both WIG and QF consistently perform well in a variety of collections. On the other hand, no matter how we tune the parameter settings of clarity and robustness on the GOV2 collection, their performance on this collection is still noticeably lower than the performance on the traditional TREC collections used in Section 4.1.

Noticing that all of our prediction models are based on the language modeling framework, why some of them are sensitive to collections while others are not? The heart of the language model technique is the estimation of the probability $P(w|M)$ where w is a language unit and M is a language model. With regard to a Web collection, the estimation of $P(w|M)$ is generally less accurate, considering the fact that Web collections are often much more heterogeneous than traditional TREC collections. For the clarity score, it directly includes $P(w|M)$ in its computation, since the clarity is defined as the sum of $P(w|M)\log\frac{P(w|M)}{P(w|C)}$ over all terms in collection C. Regarding the robustness score, in essence this method directly perturbs $P(w|M)$ if the retrieval function is based on language modeling, which is true in our experiments. However, both WIG and QF make use of $P(w|M)$ in an indirect way. For WIG, it makes use of $P(w|M)$ in the form of $\log P(w|M)$ instead of directly incorporating $P(w|M)$ in its calculation. This logistic form makes WIG more resilient to estimation noise in $P(w|M)$. With respect to QF, this technique does not directly rely on $P(w|M)$, since it uses the overlap between two rankings for performance prediction.

In short, estimation accuracy of $P(w|M)$ has a more impact on clarity and robustness than WIG and QF. We believe this is the main reason that the prediction accuracy of clarity and robustness drops noticeably on the GOV2 collection.

4.2.2 Named-Page (NP) Finding Queries

The Named-page (NP) finding task is a navigation task where a user is interested in finding a particular Web page that she may have seen before. For example, "TREC proceedings" (TREC topic # NP973) is a NP query submitted by a user who is looking for the TREC publication page that contains TREC proceedings. This task is fundamentally different from the ad-hoc retrieval task based on topic relevance. Usually there is only one correct answer for a NP queries, as opposed to multiple relevant documents for a content-based query. As we stated earlier, most research on performance prediction focuses on the ad-hoc retrieval task and we know of no published work by other researchers that explicitly addresses the problem of performance prediction for NP queries.

Regarding our prediction models, the clarity score, the robustness score and query feedback are mainly designed for content-based queries while the first rank change (FRC) is solely designed for NP queries. Only WIG is designed for handling both types of queries. Accordingly, the major goal of this section is testing the prediction power of WIG and FRC. In addition, we are also interesting in how well the techniques designed for content-based queries will perform for NP queries. To this end, we include the clarity score and the robustness score as two baselines.

We still use the GOV2 collection as our test collection. We adopt the Named-Page finding topics of the Terabyte Tracks of 2005 and 2006 and we name them TB05-NP and TB06-NP respectively. Table 4.8 summarized the above data set.

Retrieval performance of NP queries is measured by reciprocal rank of the first correct answer. Again, we make use of the Markov Random field model for retrieval and we adopt the same setting of retrieval parameters used in [51,53].

Table 4.8 : Summary of data sets for named-page finding queries

Name	Collection	Topic Number	Query Type
TB05-NP	GOV2	NP601-NP872	NP
TB06-NP	GOV2	NP901-NP1081	NP

We use the correlation with the reciprocal ranks measured by the Pearson's correlation test to evaluate prediction quality. The results are presented in Table 4.9.

To see the full potential of the clarity score, we tune it in different ways. We found that using the first ranked document to build the query model yields the best prediction accuracy. This makes sense since retrieval performance of NP queries heavily depends on the quality of the first ranked document. We also attempted to utilize document structure by using the *mixture of language* models mentioned in Section 3.4. Little improvement was obtained. The correlation coefficients for the clarity score reported in Table 4.9 are the best we have found. As we can see, WIG and FRC considerably outperform the clarity score technique on both of the runs. This confirms our intuition that the use of a coherence-based measure like the clarity score is inappropriate for NP queries.

Table 4.9 : Pearson's correlation coefficients for correlation with reciprocal ranks on the Terabyte Tracks (NP) for clarity score, robustness score, WIG, the first rank change (FRC). Bold cases mean the results are statistically significant at the 0.05 level.

Methods	Clarity	Robust	WIG	FRC
TB05-NP	**0.150**	**-0.370**	**0.458**	**0.440**
TB06-NP	0.112	-0.160	**0.478**	**0.386**

Regarding the robustness score, we also tune the parameters to see its full potential and report the best we have found. We observe an interesting and surprising negative correlation with reciprocal ranks. We explain this finding briefly. A high robustness score means that a number of top ranked documents in the original ranked list are still highly ranked after perturbing the documents. The existence of such documents is a good sign of high performance for content-based queries as these queries usually contain a number of relevant documents. However, with regard to NP queries, one fundamental difference is that there is only one relevant document for each query. The existence of such documents can confuse the ranking function and lead to low retrieval performance. Although the negative correlation with retrieval performance exists, the strength of the correlation is weaker and less consistent compared to WIG and FRC as shown in Table 4.9.

Moreover, from Table 4.9 we can see that prediction techniques like clarity score and robustness score that are mainly designed for content-based queries face significant challenges and are inadequate to deal with NP queries. Our two techniques proposed for NP queries consistently demonstrate good prediction accuracy, displaying initial success in solving the problem of predicting performance for NP queries.

Why do techniques designed mainly for content-based queries generally have difficulty in coping with NP queries? Many of these techniques are related to the famous cluster hypothesis in IR: closely-related documents tend to be relevant to the same request [58]. Assuming that this hypothesis holds, if retrieval is of high quality, top ranked documents should be roughly on the same topic and are highly related to each other. Therefore, the relationship among these top retrieved documents can be used to predict performance. For example, the clarity method directly measures ranked list coherence by comparing the query model (which can be see as a summary of top retrieved documents) to the collection model. For the robustness score, a ranked list of very dissimilar documents is sensitive to document perturbation and is very likely to have a low robustness score. This broad idea is also frequently adopted explicitly or implicitly in prediction models developed by other researchers. For instance, Vinay et al. [29] uses the Cox-Lewis statistics to measure the clustering tendency of top ranked documents for prediction. However, when it comes to NP queries, the clustering hypothesis generally does not hold any more since there is only one correct answer for NP queries. Accordingly, techniques based on the clustering hypothesis are likely to be much less effective for NP queries.

In comparison, WIG and FRC do not rely on this hypothesis. We believe this is one of the main reasons that WIG and FRC works well for NP queries. Combined with results from the previous section, we observe that WIG performs well for both types of queries, a desirable property that most prediction techniques lack.

4.2.3 Unknown Query Types

In this section, we consider a more challenging situation where the type of a given query is unknown. Specifically, we run two kinds of experiments without access to query type labels. First, we assume that only one type of query exists but the type is unknown. Second, we experiment on a mixture of content-based (CB) and NP queries. The following two subsections will report results for the two conditions respectively. Previously WIG was shown to be the only one of our models (in fact we know of no other models that claim to be effective for both CB and NP queries) that can deal with both types of queries provided that query types are known in advance. The major goal of this section is to test the prediction power of WIG under the demanding situation that no prior information on query types is available.

4.2.3.1 Only One Query Type Exists

We first consider a simple case by assuming that all queries are of the same type, that is, they are either NP queries or content-based queries. We consider two cases: (1) CB: all 150 title queries from the ad-hoc task of the Terabyte Tracks 2004-2006 , and (2)NP: all 433 NP queries from the named page finding task of the Terabyte Tracks 2005 and 2006.

We take a simple strategy by labeling all of the queries in each case as the same type (either NP or CB) regardless of their actual type. The computation of WIG will be based on the labeled query type instead of the actual type. There are four possibilities with respect to the relation between the actual type and the labeled type. The correlation with retrieval performance under the four possibilities is presented in Table 4.10. For example, the value 0.445 at the intersection between the second row and the third column shows the Pearson's correlation coefficient for correlation with average precision when the content-based queries are incorrectly labeled as the NP type.

Based on these results, we recommend treating all queries as the NP type when only one query type exists and accurate query classification is not feasible, considering the risk that a large loss of accuracy will occur if NP queries are incorrectly labeled as content-based queries. These results also demonstrate the strong adaptability of WIG to different query types.

Table 4.10 : Comparison of Pearson's correlation coefficients for correlation with retrieval performance under four possibilities on the Terabyte Tracks (NP). Bold cases mean the results are statistically significant at the 0.05 level.

	CB (labeled)	NP (labeled)
CB (actual)	**0.536**	**0.445**
NP (actual)	**0.174**	**0.467**

4.2.3.2 A Mixture of Content-based and NP Queries

An unknown mixture of the two types of queries is a more realistic description of the situation that a Web search engine faces. Considering the fact that retrieval performance of the two types of queries is measured differently, we do not use the correlation with retrieval performance to evaluate prediction accuracy for the mixed situation. Instead, we evaluate prediction accuracy by how accurately poorly-performing queries can be identified by the prediction method assuming that actual query types are unknown (but we can predict query types). This is a challenging task because both the predicted and actual performance for one type of query can be incomparable to that for the other type.

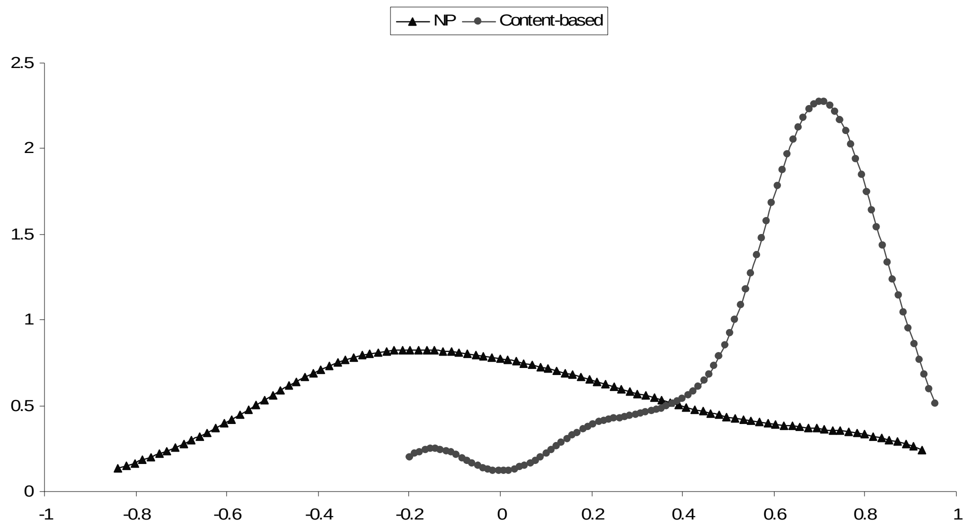


Figure 4.4: Distribution of robustness scores for NP and CB queries. The NP queries are the 252 NP topics from the 2005 Terabyte Track. The content-based queries are the 150 ad-hoc title from the Terabyte Tracks 2004-2006. The probability distributions are estimated by the Kernel density estimation method.

Next we discuss how to implement our evaluation. We create a query pool which consists of all of the 150 ad-hoc title queries from Terabyte Track 2004-2006 and all of the 433 NP queries from Terabyte Track 2005&2006. We divide the queries in the pool into classes: “good” (better than 50% of the queries of the same type in terms of retrieval performance) and “bad” (otherwise). According to these standards, a NP query with the reciprocal rank above 0.2 or a content-based query with the average precision above 0.315 will be considered as good.

Then, each time we randomly select one query Q from the pool with probability p that Q is content-based. The remaining queries are used as training data. We first decide the type of query Q according to a query classifier. Namely, the query classifier tells us whether query Q is NP or content-based (CB). Based on the predicted query type and the score computed for query Q by a prediction technique, a binary decision is made about whether query Q is good or bad by comparing to the score threshold of the predicted query type obtained from the training data. Prediction accuracy is measured by the accuracy of the binary decision. In our implementation, we repeatedly take a test query from the query pool and prediction accuracy is computed as the percentage of correct decisions, that is, a good(bad) query is predicted to be good (bad). It is obvious that random guessing will lead to 50% accuracy.

Let us take the WIG method for example to illustrate the process. Two WIG thresholds (one for NP queries and the other for content-based queries) are trained by maximizing the prediction accuracy on the training data. When a test query is labeled as the NP (CB) type by the query type classifier, it will be predicted to be good if and only if the WIG score for this query is above the NP (CB) threshold. Similar procedures will be taken for other prediction techniques.

Table 4.11: Comparison of prediction accuracy for five strategies in the mixed-query situation. Two ways to sample a query from the pool: (1) the sampled query is content-based with the probability p=0.6. (that is, the query is NP with probability 0.4) (2) set the probability p=0.4.

Strategies	Robust	WIG-1	WIG-2	WIG-3
p=0.6	0.565	0.624	0.665	0.684
P=0.4	0.567	0.633	0.654	0.673

Now we briefly introduce the automatic query type classifier used in this paper. We find that the robustness score, though originally proposed for performance prediction, is a good indicator of query types. The use of the robustness score for query classification is motivated by the observation obtained from our previous prediction experiments reported in Section 4.2.1 and 4.2.2 that the robustness score behaves very differently between these two types of queries. We find that on average content-based queries have a much higher robustness score than NP queries. For example, Figure 4.4 shows the distributions of robustness scores for NP and content-based queries. According to this finding, the robustness score classifier will attach a NP (CB) label to the query if the robustness score for the query is below (above) a threshold trained from training data. In addition, many other techniques have been proposed for the task of query classification [65][66]. For example, Kang et al.[66] used features such as POS (part-of-speech), term distributions and anchor text for classification. We expect that further improvement on classification accuracy can be achieved when the robustness score is used in combination with other features.

We consider four strategies in our experiments. In the first strategy (denoted by "*robust*"), we use the robustness score for query performance prediction with the help of a perfect query classifier that always correctly map a query into one of the two categories (that is, NP or CB). This strategy represents the level of prediction accuracy that current prediction techniques can achieve in an ideal condition that query types are known. In the next following three strategies, the WIG method is adopted for performance prediction. The difference among the three is that three different query classifiers are used for each strategy: (1) the classifier always classifies a query into the NP type, (2) the classifier is the robust score classifier mentioned above, (3) the classifier is a perfect one.   These three strategies are denoted by *WIG-1*, *WIG-2* and *WIG-3* respectively. The reason we are interested in WIG-1 is based on the results from section 4.3.1.

The results for the four strategies are shown in Table 4.11. For each strategy, we try two ways to sample a query from the pool: (1) the sampled query is CB with probability p=0.6. (the query is NP with probability 0.4)   (2) set the probability p=0.4. From Table 8 We can see that in terms of prediction accuracy WIG-2 (the WIG method with the automatic query classifier) is not only better than the first two cases, but also is close to WIG-3 where a perfect classifier is assumed. Some further improvements over WIG-3 are observed when combined with other prediction techniques. The merit of WIG-2 is that it provides a practical solution to automatically identifying poorly performing queries in a Web search environment with mixed query types, which poses considerable obstacles to most prediction techniques.

4.2.4 Prediction on Realistic Web Data

In the previous section, we see the superiority of WIG over other methods in Web search environments. However, our evaluation was performed under laboratory settings which consist of carefully-selected topic sets with relevance judgments made by human assessors and pre-defined retrieval tasks. In this section, we focus on testing the real-world performance of WIG by experimenting on realistic Web data gathered from a commercial search engine.

Specifically, the dataset used in this section is a query log file that contains about 149 million queries collected by a web search company during of one month period (from May 1 to May 31 of 2006). For each query, the following information is associated: (1) query ID,(2) the time the query is submitted, (3) the content of the query, (4) the result(s) clicked by the user who submits the query, (5) the number of returned results. Notice that (4) and (5) are not available if no results are clicked for the query. No relevance information on queries is available. This dataset represents the kind of information that a typical web search engine can readily obtain from user interaction.

Considering the fact that positive correlation between WIG scores and retrieval performance was observed previously, it is natural to investigate whether a similar relationship exists for WIG on the dataset described above. One problem with this is that we can not calculate the actual retrieval performance without any relevance judgments. Instead of relying on human relevance judgments that are accurate but usually too expensive to obtain in a Web collection, we resort to click information that may be noisy but can provide valuable information about relevance. We assume that a document clicked by the user can be roughly viewed as relevant to her information need. Accordingly our hypothesis is that high WIG scores would predict more clicks on search results. That is, WIG is related to user's preference for search results.

We now describe our experimental design to test the above hypothesis. We randomly sample 2000 queries from the query log file. We assume that each query is issued by a unique user (that is, one-to-one correspondence between a query and a user) .We divide these queries into three groups according to the number of results the user clicked when seeing the ranked list of results in response to her query. Table 4.12 gives the details. For example, Group A represents those users who do not click any of the returned results. In fact, group A, B and C represent three levels of user interest in search results in ascending order. The percentage of each group in our sampled query set is also provided in Table 4.12. We can see that the majority of users only click one of the retrieved results.

For each query in the sampled query set, we compute the WIG score. For WIG calculation, in addition to the query itself we need the ranked list in response to the query and a collection. We use the provided search engine API to download the top ranked documents for the query. The GOV2 collection is used to approximate the Web collection statistics required in WIG calculation. The parameter settings of WIG are the same as used in Section 4.2.2. The distributions of WIG scores for the three groups are presented in Table 4.13. We adopt two statistics to represent the distribution of WIG scores for each group: sample mean and sample variance. The size of each group (the number of queries in the group) is also provided.

Table 4.12 : Division of test queries into three groups based on the number of clicked results

Group	A	B	C
# of clicked results	0	=1	>=2
Percentage	34.8%	50.1%	14.1%

Table 4.13 : Distributions of WIG scores for Group A, B and C

Group	A	B	C
Sample mean of WIG score	5.340	6.040	6.648
Sample variance of WIG score	11.645	8.120	8.877
Size	695	1002	283

Let $WIG(A), WIG(B)$ and $WIG(C)$ represent the mean of WIG scores in group A,B and C respectively, that is , $WIG(A) = \frac{1}{|A|}\sum_{q \in A} WIG(q)$ where $|A|$ is the size of group A and $WIG(q)$ is the WIG score for query q. $WIG(B)$ and $WIG(C)$ have similar definitions. From Table 4.13 we observe that $WIG(C) > WIG(B)$ and $WIG(B) > WIG(A)$. Further investigation shows that both of the two differences ($WIG(C)$ - $WIG(B)$ and $WIG(B)$ - $WIG(A)$) are statistically significant at the 95% confidence level according to the student t test [61]. This shows that high WIG scores suggest more clicks, which is consistent with our previous finding that high WIG scores generally correspond to high retrieval performance.

We want to point out the correlation between WIG and clicks is not strong enough to make a conclusion that WIG alone can accurately predict clicks, considering the large variance of WIG scores in each group as shown in Table 4.13. This is due to the fact that user preferences for search results depend on many factors other than retrieval quality. For example, user education background may have a huge impact on their preferences. In fact, we observe in the dataset that for the same query some users do not click on any of the returned results while others do click. Since WIG is an effective feature only for predicting relevance and clicks are only related to relevance to some degree, we do not expect that clicks can be accurately predicted by WIG alone. However, WIG is still a useful feature as shown in the above experiment.

4.3 Combination of performance predictors

We observe that our predictors can sometimes perform better when linearly combined, due to the fact that they capture different aspects of the retrieval process that have a major impact on retrieval effectiveness. For example, the Pearson's correlation coefficients for the clarity score and the robustness score on Robust 04 are 0.507 and 0.557 respectively according to Table 4.2. When we combine the two predictors by a simple linear combination, the corresponding Pearson's correlation coefficient is increased to 0.613 [7]. In Web search environments, a similar trend is observed in our paper [68]: the linear combination of WIG and QF is better than WIG alone for content-based queries and the linear combination of WIG and FRC is better than WIG alone for named-page finding queries. Other researchers also report that the combination of multiple prediction features can provide better prediction accuracy than anyone when used in isolation [25]. In general, performance prediction should be done using a combination of resources, if this is computationally possible.

4.4 Model Comparison for Query Expansion Prediction

In this section, we address the task of query expansion prediction. The major goal of this section is to test the ability of model comparison in sensing poor expansion results. Specifically, we perform both an unexpanded retrieval and an expanded retrieval. We adopt the query likelihood model [41] for our unexpanded retrieval for each query. For our expanded retrieval, we use the relevance model [40] which is a conceptually simple and effective way for implementing query expansion. We will explore if model comparison can accurately predict the change in performance between the two retrievals for a particular query.

We apply the model comparison method for the task of query expansion prediction on a variety of data sets as shown in Table 4.14. All queries used in this section are titles of TREC topics. Regarding the calculation of model comparison score, we set T to 10 in Equation 3.23. (Tests show that this method is not very sensitive to T as long as it is in the range of 5 to 50).

Table 4.14 : Summary of test collections for query expansion prediction

TREC	Collection	Topic Number	Number of Document
1+2+3	Disk 1+2+3	51-150	1,078,166
5	Disk 2+4	201-250	524,929
Robust 2004	Disk 4+5 minus CR	301-450; 601-700	528,155
Terabyte 04-05 (ad-hoc task)	GOV2	701-800	25,205,197

Figure 4.5 shows our models comparison scores applied to the Robust Track 04 data. The delta average precision is the value with relevance retrieval minus the value with query-likelihood retrieval. Highly positive comparison scores are an indicator that the performance of the expanded retrieval may be significantly worse than the unexpanded retrieval. Moreover, highly positive scores, when they occur, are often well separated from the other scores. This separation makes it possible to distinguish hard-to-expand queries from others by setting a threshold. Specifically, after setting a threshold of model comparison score, we use expanded retrieval for a given query if the model comparison score is below the threshold and vice versa. We call this strategy *selective query expansion based on model comparison* (SQEMC for short).

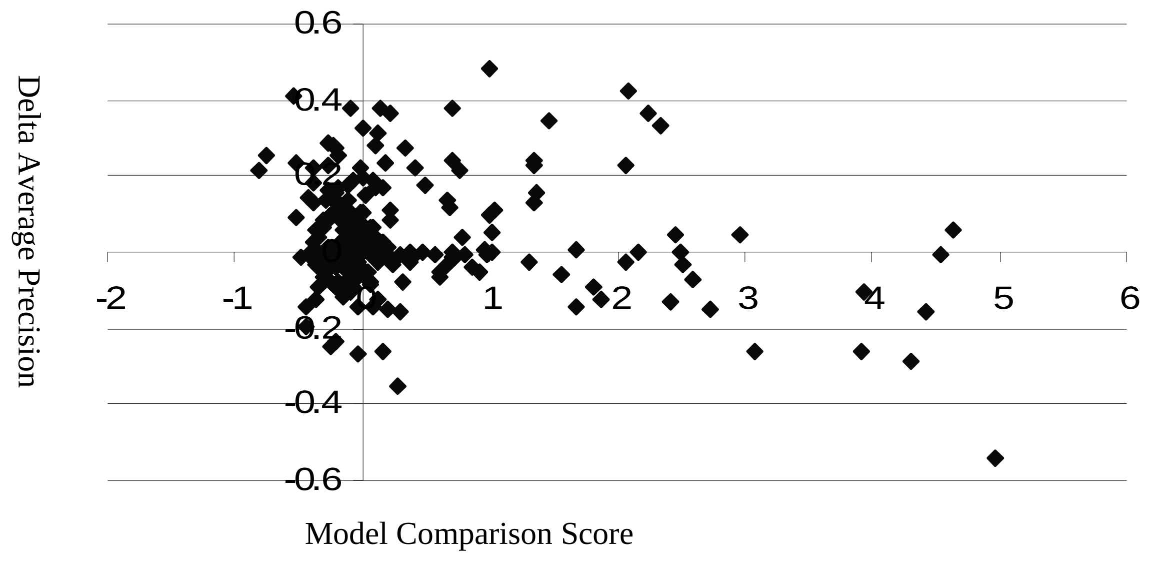


Figure 4.5 a scatter plot of model comparison scores versus delta average precision for each of the 250 queries of Robust 2004.The delta average precision is the value with relevance retrieval(REL) minus query-likelihood retrieval(QL).

Table 4.15 : Mean Average Precision
QL: query-likelihood
REL: query expansion by using the relevance model
SQEMC: selective query expansion based on model comparison by predicting change between QL and REL in average precision using a threshold learned from training data

TREC	QL	REL	SQEMC
TREC123	0.187	0.249	0.245
TREC5	0.149	0.161	0.168
Robust 04	0.244	0.280	0.286
Terabyte 04-05	0.291	0.311	0.313

Table 4.15 shows the mean average precision for SQEMC compared to using relevance model retrieval (REL) and query-likelihood retrieval (QL). We obtain the

results for SQEMC by leave-one-out cross-validation, by setting the threshold score using training data, then applying the threshold to held-out test data. Generally we do not observe large mean average precision improvement over relevance retrieval. The main reason is that this method is only capable of detecting a small percentage of queries that perform very poorly on expansion as shown in Figure 4.5. Moreover, SQEMC usually does not hurt retrieval performance.

Examining some example queries is illuminating and can help us understand the strength and weakness of this method. We first look at cases when model comparison works well. In the case of “supercritical fluid” [delta average precision (REL-QL): -0.290, model comparison score: 4.313], loosing the requirement that documents use the exact query terms frequently to be highly ranked (by going from unexpanded retrieval to expanded retrieval) ranks documents too highly that do not contain all the correct jargon term frequently. In other words, in this case matching the exact technical terms is what satisfying the information need requires. This query receives a high model comparison score because query expansion makes the technical terms occur less frequently in the search results. In the example of the “tourist violence” query [delta average precision (REL-QL): 0.254, model comparison score:-0.757], broadening the search to contain closely related terms can help. However, there are hard-to-expand queries that the method fail to detect. One is “Legionnaires disease” [delta average precision (REL-QL): -0.248, model comparison score:-0.256] where documents can contain the terms “legionnaire (meaning soldier)” and “disease” (and related words) yet not be about Legionnaires' disease, leading to a low comparison score despite its hard-to-expand status.

4.5 Summary

Several prediction techniques (clarity, ranking robustness, query feedback and WIG) were evaluated for the task of performance predication. Our major findings are:

(1) Regarding the clarity score, we found that it performs reasonable well for content-based queries on traditional TREC test collections. However, the prediction accuracy of clarity drops remarkably in a Web search environment.

(2) For the ranking robustness technique, there are two variants: robustness score for content-based queries and first rank change (FRC) for named-page (NP) finding queries. FRC is found to be effective for NP queries on the GOV2 collection. Like the clarity score, the performance of robustness score for content-based queries is good on traditional TREC test collections but is low on the GOV2 web collection. One interesting phenomenon we observed about the robustness score is a moderate negative correlation with retrieval performance for NP queries. Though the correlation is not strong, the opposite behavior of the robustness score between the two types of queries motivates us to investigate the possibility of the use of robustness score for query classification. Our results show that the robustness score is a good feature for distinguishing between the two query types.

(3) Query feedback (QF) is found to be effective for content-based queries on both traditional TREC collections and the GOV2 collection.

(4) WIG offers consistent prediction accuracy across various search scenarios. WIG provides a uniform framework to deal with both content-based and named-page finding queries. In fact, as far as we know, WIG is the only predictor we have found so far that can successfully deal with both types of queries, making it particularly suitable for performance prediction in a Web search environment. In addition, with the help of an automatic query classifier, WIG offers a practical solution to predicting mixed-query performance. Our experiment on realistic Web data collected from a commercial Web search engine shows a tendency that high WIG scores predict more clicks on search results. We also find that description queries can make WIG less effective compared to title queries.

(5) For the clarity score, the robustness score and QF, we observe no noticeable change of prediction accuracy from short queries based on the topic title to long queries based on the topic description.

In addition, we proposed a method called *model comparison* to address the task of query expansion prediction. We found that this method can improve retrieval consistency by catching a small proportion of queries and avoiding some poor expansions, although no significant improvement over expansion retrieval was observed in terms of mean average precision.

CHAPTER 5

IMPLEMENTATION OF PREDICTION MODELS

We evaluated the effectiveness of our models in the previous chapter. Other than effectiveness, efficiency is another major concern especially when facing a large amount of data. Accordingly, we devote this chapter to discussing the implementation of our prediction models.

(1) Clarity Score

Let us revisit the definition of clarity which is given in Equation 3.5:

$$clarity\,score = \sum_{w \in V} P(w \mid Q) \log_2 \frac{P(w \mid Q)}{P_{coll}(w)}$$

The collection model $P_{coll}(w)$ is typically estimated by the relative frequency of term w in the collection. The estimation of $P_{coll}(w)$ only needs two kinds of statistics: the number of occurrences of term w in the collection and the total number of terms in the collection. Since almost all indexes will store these statistics, the computation of $P_{coll}(w)$ costs nearly no additional space and time. In fact, the most time-consuming and space-consuming part of clarity computation is the estimation of the query model $P(w \mid Q)$ which is estimated as follows:

$$P(w \mid Q) = \sum_{D} P(w \mid D) P(D \mid Q)$$

Theoretically, the D in the above equation includes every document in the collection. This means that we need to compute and store the probability $P(w \mid Q)$ for every word in the collection, which is almost infeasible when the collection is large. One practical solution is that we perform a first round retrieval and limit the D to the top K ranked documents from the retrieval. By doing so, we only need to pay attention to terms that do occur in the top K ranked documents. However, both time and space complexity can still be high even after this approximation. In essence, the query model $P(w|Q)$ can be viewed as a type of automatic query expansion (AQE) which is usually slow in practical applications. Methods for speeding up AQE can be helpful for improving the efficiency of clarity computation.

(2) Ranking Robustness

The efficiency of both robustness score and FRC is dominated by the following two steps: i) perturb term counts in a document, ii) re-rank the perturbed documents. If only query terms are considered, the perturbation process (the first step) can be implemented efficiently. For example, we can create a table where entry (i,j) records the number of occurrences of query term j in document i. Since a cutoff is set on the ranked list and the query is usually relatively short, the size of the table is small and therefore can be stored in main memory. The perturbation process can be directly performed on this table by perturbing the term count in each cell of the table. Regarding the second step, the re-ranking process will be fast with the help of the perturbed table, since this table can be used as an index of the perturbed documents to be re-ranked.

(3) Weighted information gain

WIG can be implemented very efficiently. The calculation of WIG is given in Equation 3.20. We rewrite it as follows:

$$WIG(Q_i, C, L) = \frac{1}{K} \sum_{D_t \in T_K(L)} \sum_{\xi \in F(Q_i)} \lambda_\xi \log \frac{P(\xi \mid D_t)}{P(\xi \mid C)}$$
$$= [\frac{1}{K} \sum_{D_t \in T_K(L)} \sum_{\xi \in F(Q_i)} \lambda_\xi \log P(\xi \mid D_t)] - \sum_{\xi \in F(Q_i)} \lambda_\xi \log P(\xi \mid C)$$
$$let\ score(Q_i, D_t) = \sum_{\xi \in F(Q_i)} \lambda_\xi \log P(\xi \mid D_t), score(Q_i, C) = \sum_{\xi \in F(Q_i)} \lambda_\xi \log P(\xi \mid C)$$
$$WIG(Q_i, C, L) = [\frac{1}{K} \sum_{D_t \in T_K(L)} score(Q_i, D_t)] - score(Q_i, C) \qquad (5.1)$$

In Equation 5.1, $score(Q_i, D_t)$ is exactly the relevance score of document D_t for query Q_i when the Markov Random Field (MRF) model is adopted for retrieval. Similarly, $score(Q_i, C)$ is the relevance score of collection C by treating the collection as a whole. For $score(Q_i, D_t)$, we can directly copy the corresponding document score from the given ranked list when the MRF model is used for retrieval. Even if another retrieval technique is adopted, the computation of $score(Q_i, D_t)$ is still quite efficient since only features occurring in query Q_i are considered. Regarding $score(Q_i, D_t)$, $P(\xi \mid C)$ is usually estimated by the relative frequency of feature ξ in the collection. This information can be readily obtained from the index. Considering that both single term and phrase features are used in the computation of WIG, we require that the index supports both types of features. The Indri search engine [62] is one example that fully supports such a request. In short, the computation of WIG can be made extremely efficient under the proper conditions.

(4) Query feedback (QF)

There are two major bottlenecks for the implementation of QF: i) the estimation of the ranked list language model , ii) an extra retrieval required for measuring the similarity between the original query and the new query. The ranked list model is a variant of the query model used in the computation of clarity. Therefore, the efficiency issues of implementing the query model in clarity calculation also apply here. Both i) and ii) make it difficult to implement QF efficiently.

(5) Model Comparison

The most complex part for computing model comparison scores comes from the estimation of the ranked list model which is stated above.

Based on the above discussion, we can see that the implementation of WIG is the most efficient among our prediction models. Considering the fact that WIG shows satisfactory prediction accuracy across a variety of search settings (especially in Web search environments), WIG is our first choice for the task of performance prediction in realistic applications involving a large amount of data.

CHAPTER 6

PREDICTING DOCUMENT QUALITY FOR WEB AD-HOC RETRIEVAL

6.1 Overview

In this chapter, we address the problem of document quality and its application to Web retrieval. As we mentioned before, this problem can be viewed as performance prediction at the level of individual documents. To achieve the goal of improving the performance of Web ad hoc retrieval by exploiting document quality information, we propose a document quality model that incorporates features other than link structure. This quality model is incorporated into the basic query likelihood retrieval model in the form of a prior probability. We first introduce two quality metrics, that is, information-to-noise ratio and collection-document distance. The latter is a novel feature found to be helpful for identifying low quality documents. We then show how to estimate the quality of a web document using a naïve Bayes classifier which makes use of these two features. This naïve Bayes classifier is embedded as a prior probability in the query likelihood model. We evaluate our document quality model on three TREC web collections (GOV2, WT2G and WT10G) in terms of three measures: precision in the top ranked documents, mean average precision and MRR. Our results demonstrate that, on average, the retrieval model incorporating quality is significantly better than the baseline in terms of MRR and precision at the top ranks, although the impact of our model on mean average precision is quite small. Last, we give a detailed query analysis to understand the limitations of our model.

6.2 Quality Metrics

We focus on two quality metrics, collection-document distance and information-to-noise ratio, the first of which is new and the second having been used with some success in a previous study [39].

6.2.1 Collection-document Distance

The Collection-Document Distance (CDD for short), is simply the relative entropy, or Kullback-Leibler (KL) divergence, between the collection and document unigram language models. The collection or background language model is estimated using the word occurrence frequencies over the whole collection (e.g. GOV2).

Given a document D and a collection C, the CDD is given by

$$CDD = \sum_{w} P_{coll}(w \mid C) \log \frac{P_{coll}(w \mid C)}{P_{doc}(w \mid D)} \qquad (6.1)$$

$$where \quad P(w \mid D) = \lambda P_{doc}(w \mid D) + (1-\lambda) P_{coll}(w \mid C)$$

$$P_{doc}(w \mid D) = \frac{\#Count(w, D)}{\| D \|}, P_{coll}(w \mid C) = \frac{\#Count(w, C)}{\| C \|}$$

In this formulation, we use linear smoothing for estimating the document language model probabilities.

Our hypothesis is that low quality documents will have unusual word distributions. In other words, if a document differs significantly from the word usage in an average document, the quality of this document may be low. In the CDD measure, the average document is represented by the collection language model. The KL divergence between the collection language model and the document language model (i.e. the CDD) indicates how different these distributions are. The higher the CDD is, the more unusual the word distribution of the document is, and the more likely, according to our hypothesis, that the document is of low quality.

Let us consider three cases that are helpful for understanding why CDD can predict low quality documents.

Case 1: documents that are tables or lists. Common words, such as pronouns, adjectives and verbs, would have very low numbers of occurrences, which makes the document language models quite different from the collection language model.

Case 2: documents that have misspelled words. The probability of a misspelled word in the collection is much lower than that of normal words. If any document contains misspelled words, the CDD tends to be high.

Case 3: documents where the frequency of some term is unnecessarily high. Since the web environment contains competing profit seeking ventures, one may intentionally increase the occurrence of some keywords in a document to get attention. CDD can recognize this case.

As an alternative to Equation 6.1, one can compute the divergence with the role of the collection language model and the document language model reversed. The method shown in Equation 6.1 performs slightly better in our evaluations and is used throughout this thesis. The value of the parameter λ is determined empirically and is 0.8 for all runs in this paper.

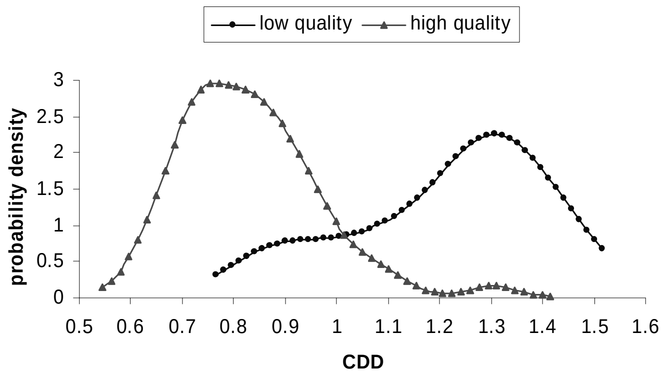


Figure 6.1: Distribution of CDD values for low and high quality documents.

Fig 6.1 shows the distributions of CDD values for high quality and low quality documents respectively. (The details on the data used to generate this figure will be described in section 6.3.1.) These two distributions are estimated from our training data by the Kernel density estimation method that will be discussed later. We can see that there is an obvious separation between the two classes of documents.

6.2.2 Information-to-noise ratio

The information-to-noise ratio is computed as the total number of terms in the documents after indexing divided by the raw size of the document [39]. This metric predicts low quality documents based on a different characteristic than the CDD metric. Consider a web document that has only a few words and many HTML tags which will be removed after indexing. The information-to-noise ratio of this document is very low and the quality of this document also tends to be low.

Fig 6.2 shows the distributions of information-to-noise ratios for high quality and low quality documents respectively. (The details on the data used to generate this figure will be described in section 6.3.1.) The two distributions are also estimated from our training data by the Kernel density estimation method. As we can see, a document with a low information-to-noise ratio is much more likely to be of low quality.

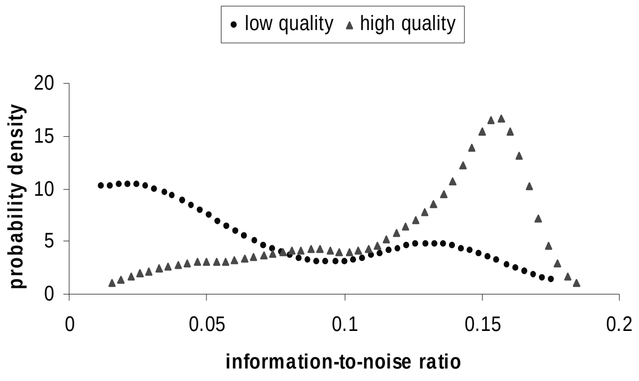


Figure 6.2: Distribution of information-to-noise ratios for low and high quality documents.

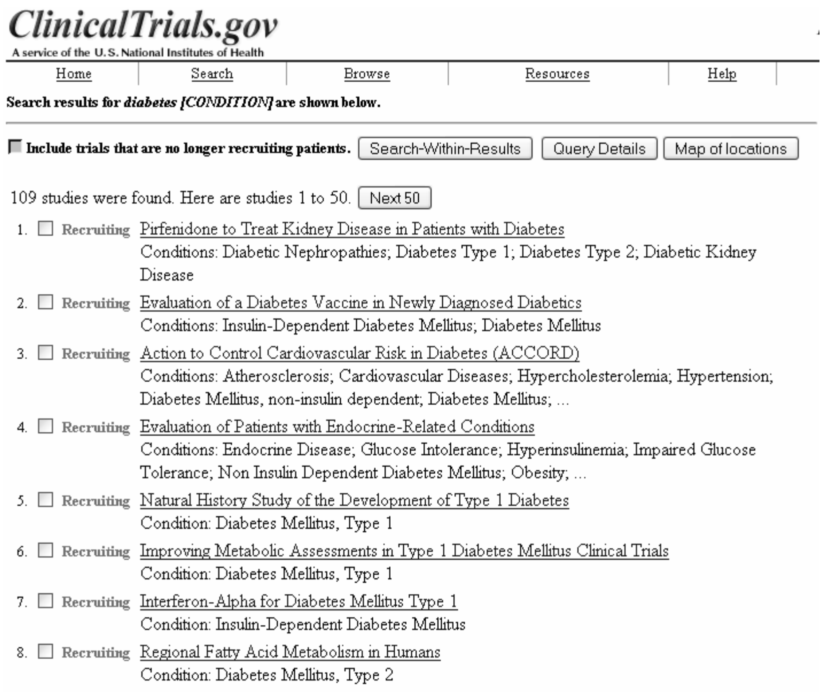
6.3 Predicting Document Quality

In this section, we show how to estimate the quality of a Web document by a naïve Bayes classifier using the two quality metrics described in the previous section.

6.3.1. Training data

First of all, we give the details of how the training data were created. These data are used for estimating the parameters in our naïve Bayes classifier.

We ran 50 title queries (TREC topics 701-750) from the 2004 Terabyte Track on the GOV2 collection. The search algorithm used is the query-likelihood model with Dirichlet smoothing [41]. We looked at the top ten retrieved documents for each query (that is, 500 documents in total). We manually judged these documents either as high quality or low quality. These labeled documents will be used as the training data in our experiments described in section 6.5. In the experiments involving Terabyte Track 2004, we used five-fold cross validation to avoid testing on the training data.



Search results for *diabetes [CONDITION]* are shown below.

Include trials that are no longer recruiting patients.

109 studies were found. Here are studies 1 to 50.

1. Recruiting Pirfenidone to Treat Kidney Disease in Patients with Diabetes
   Conditions: Diabetic Nephropathies; Diabetes Type 1; Diabetes Type 2; Diabetic Kidney Disease
2. Recruiting Evaluation of a Diabetes Vaccine in Newly Diagnosed Diabetics
   Conditions: Insulin-Dependent Diabetes Mellitus; Diabetes Mellitus
3. Recruiting Action to Control Cardiovascular Risk in Diabetes (ACCORD)
   Conditions: Atherosclerosis; Cardiovascular Diseases; Hypercholesterolemia; Hypertension; Diabetes Mellitus, non-insulin dependent; Diabetes Mellitus; ...
4. Recruiting Evaluation of Patients with Endocrine-Related Conditions
   Conditions: Endocrine Disease; Glucose Intolerance; Hyperinsulinemia; Impaired Glucose Tolerance; Non Insulin Dependent Diabetes Mellitus; Obesity; ...
5. Recruiting Natural History Study of the Development of Type 1 Diabetes
   Condition: Diabetes Mellitus, Type 1
6. Recruiting Improving Metabolic Assessments in Type 1 Diabetes Mellitus Clinical Trials
   Condition: Diabetes Mellitus, Type 1
7. Recruiting Interferon-Alpha for Diabetes Mellitus Type 1
   Condition: Insulin-Dependent Diabetes Mellitus
8. Recruiting Regional Fatty Acid Metabolism in Humans
   Condition: Diabetes Mellitus, Type 2

Figure 6.3: "Low quality" document retrieved in response to the query *controlling type II diabetes*.

Document quality is an inherently subjective concept and involves many aspects such as popularity, authority and quality of writing. Since we focus on the ad-hoc content-based retrieval task, we used the following criterion for judging a document to be low quality: A document is judged as low quality if it contains few or none of the typical sentences that would be required to describe a topic. Any other document that is not judged as low quality would be regarded as high quality. In practice, most low quality documents we found consisted of primarily tables or lists. Figure 1 gives an example of part of a typical low quality document in the training data. The document contains a list of diabetes studies that are recruiting patients for trials and was retrieved in response to the query “controlling type II diabetes”.

The intuition behind this basis for quality judgments is that a relevant document for the TREC ad hoc task usually explains or describes some topic using sentences with typical English structure and vocabulary. Therefore, documents like tables or lists are unlikely to be relevant for ad hoc queries.

We examined the relationship between relevance and document quality and Table 6.1 shows the distribution of relevant documents over two classes: high quality and low quality documents.

Table 6.1 : Distribution of relevant documents in the training data

	Relevant	Non-relevant
High quality	238	171
Low quality	9	82

As we can see, the proportion of relevant documents among low quality documents is much lower than that in high quality ones. Overall, based on our training data, successfully recognizing low quality documents should be helpful for improving retrieval performance.

6.3.2 Naïve Bayes Classifier

The Naïve Bayes classifier technique is based on the Bayesian theory and assumes independence in features. Despite its simplicity, Naïve Bayes classifiers often work much better in many real applications than might be expected from their simple design [59].

In this thesis, we use this technique to predict document quality. Specifically, let D denote a document. Note that we assume that all documents belong to one of the two classes: high quality and low quality. Let H denote the high quality class, L denote the low quality class, X denote a vector of quality metric values, and π_H and π_L denote the prior probabilities of the high quality class and the low quality class respectively. Let f_H and f_L denote the probability density functions of the high quality class and the low quality class respectively. By Bayes rule , we have:

$$\Pr(D = H \mid X = x_0) = \frac{\pi_H f_H(x_0)}{\pi_H f_H(x_0) + \pi_L f_L(x_0)} \qquad (6.2)$$

Given multiple features (in this case, quality metric values) it is common to assume independence among the features. In fact, we examined the training data and found there is little correlation between the two metrics. Under this assumption, we have

$$f_j(X) = f_j(x_0) f_j(x_1), \ \ j = H, L \quad (6.3)$$

where x_0 is the CCD metric and x_1 is the information-noise ratio.

The key part of computing Pr(D=H|X) is the estimation of the probability density functions in Equation 6.2, since π_H and π_L can be simply estimated by the relative frequencies in the training data. However, it is not easy to estimate these functions since we do not know what distribution the two metrics actually follow. Instead, we adopt Kernel density estimation which does not assume any specific distribution on the features we want to estimate. Kernel density estimators belong to a class of estimators called *non-parametric* density estimators that have no fixed structure and depend upon all data points to reach an estimate.

Assume we have a random sample x_1, x_2, …x_N drawn from a probability density function *f(x)* and we wish to estimate *f(x)* at a point x_0 , the Kernel density estimator for *f(x)* at the point x_0 is defined as [59]:

$$\hat{f}(x) = \frac{1}{N\lambda}\sum_{i=1}^{N} K_\lambda(x_0, x_i) \quad (6.4)$$

where λ is the bandwidth and K_λ is a Kernel function. In this paper we use the Gaussian Kernel and Equation 6.4 can be written as

$$\hat{f}(x) = \frac{1}{N\lambda\sqrt{2\pi}} \sum_{i=1}^{N} \exp\left(-\frac{(x_i - x_0)^2}{2\lambda^2}\right) \quad (6.5)$$

There is a standard way to select the bandwidth (λ) based on minimizing the expected square error between the estimated density and the original density [59]. In this proposal, we adopt this method to calculate λ.

6.4 Document Quality Language Model

To utilize our quality predictor to improve retrieval performance, we propose the document quality language model [60] that is built on the top of the basic query likelihood model by incorporating document quality prediction in the form of a prior probability.

Specifically, given a query Q and a document D, let P(D|Q) be the probability that D is relevant given Q, the document quality language model is as follows:

$$P(D \mid Q) \propto P(Q \mid D) P(D = H \mid X)$$

where P($Q|D$) is the query likelihood model described in [41] and P(D=$H|X$) computed by Equation 6.2 can be interpreted as the document prior probability that reflects prior knowledge about the relevance of the document D[38].

6.5 Results

In this section, we present the results of comparisons between the document quality model and the query likelihood model on the three Web test collections. The details of these test collections are shown in Table 6.2. Three metrics are used for evaluation: precision at top retrieved documents, mean average precision and MRR (mean reciprocal rank). Our results show that the document quality model significantly outperforms the baseline in the evaluation using precision at top ranked documents and MRR, although the differences of MAP between the two models are quite small.

For query likelihood retrieval, we use Dirichlet smoothing with a smoothing parameter of 2500 for all runs.

Table 6.2: Summary of test collections

Test Collection	Topic Number	Number of Document
WT10G	501-550	1,692,096
WT2G	401-450	247,491
GOV2 (Terabyte Track 04-05)	701-800	25,205,197

6.5.1 Results for Precision at Top Ranks

In a typical Web search environment, few people would look at more than the first ten or twenty results.   Precision at the top ranks is a very important metric since it reflects the concern with high retrieval accuracy. In this thesis, we evaluate precision at 4 rank levels: 5, 10, 15 and 20.

Table 6.3 shows the precisions at top ranks on the GOV2 collection. To better compare our model with the baseline, all queries are divided into three types: “Pos”, “Neg” and “Eq”, which means our model is better, worse or equal to the baseline respectively. The last column in Table 6.3 shows the numbers of the three types of queries.

Table 6.3: Precision on the GOV2 collection. "Pos" means result is better than the baseline, "Neg" means result is worse than the baseline, "Eq" means result is the same as the baseline

Precision @	Query-likelihood model	Document quality model	Pos	Neg	Eq
5 docs	0.5592	0.6037	27	11	61
10 docs	0.5430	0.5673	28	12	59
15 docs	0.5207	0.5550	37	18	44
20 docs	0.5066	0.5400	38	18	43

.

Table 6.4: Precision on the WT2Gcollection

Precision @	Query-likelihood model	Document quality model	Pos	Neg	Eq
5 docs	0.4960	0.5240	9	3	38
10 docs	0.4640	0.4760	10	4	36
15 docs	0.4107	0.4280	10	3	37
20 docs	0.3880	0.3920	10	7	33

We can see that the document quality model consistently outperforms the baseline at all of the 4 rank levels. On the other hand, the majority of the queries are not affected by the quality-based prior. One reason is that high quality documents, where the differences in the prior probabilities tend to be negligible, consist of a large part of the whole collection. In other words, our model can make a difference only when there are enough low quality documents in a rank list. Approximately twice as many queries are improved by this technique than are hurt.

Table 6.5: Precision on the WT10G collection

Precision @	Query-likelihood model	Document quality model	Pos	Neg	Eq
5 docs	0.3440	0.3640	9	6	35
10 docs	0.3000	0.3240	13	5	32
15 docs	0.2880	0.2907	13	12	25
20 docs	0.2660	0.2900	19	9	22

The results for WT2G and WT10G are shown in Table 6.4 and 6.5 respectively. As with the results on the GOV2 collection, the document quality model consistently improves precision. Moreover, considering the limited size of the training data, we believe that performance could be further improved by including more training data from a variety of web collections.

In summary, these results suggest that incorporating document quality information can significantly improve precision at the top ranks.

6.5.2 Mean Average Precision Results

Mean average precision (MAP for short) is the most frequently used measure for ad hoc retrieval. In our view, MAP is a less important measure than precision at the top ranks for a typical web user, but to fully evaluate and understand the quality model, we include this measure.

Table 6.6 shows the mean average precision on GOV2, WT2G and WT10G. Percentage improvements with respect to the baseline are also given. “Pos”, “Neg” and “Eq” have the same meaning mentioned in Section 6.5.1. As we can see, although the differences between the two models are small, the document model is consistently better than the baseline on all of the three collections.

Table 6.6: MAP on the three test collections

Collection	Query-likelihood model	Document quality model	Pos	Neg	Eq
GOV2	0.2882	0.2905 (+0.8%)	55	45	0
WT2G	0.3135	0.3220 (+2.7%)	34	16	0
WT10G	0.1831	0.1840 (+ 0.5%)	27	21	2

6.5.3 MRR Results

Table 6.7 shows MRR on the three test collections. Since the document quality model is consistently better than the baseline in terms of precision at top ranked documents, it is not surprising that our model performs better on GOV2 and WT2G. WT10G is an exception where our model is slightly worse than the baseline. We discuss this more in the next section, but MRR is more sensitive to a small fluctuation in the rank list than precision. For example, if the rank of the first relevant document is second instead of first, the MRR drops from 1.0 to 0.5 while the precision in the top 5 documents may change very little.

Table 6.7 : MRR on the three test collections

Collection	Query-likelihood model	Document quality model	Pos.	Neg.	Eq.
GOV2	0.7391	0.7927	25	6	68
WT2G	0.7406	0.7781	9	1	40
WT10G	0.6215	0.6139	9	8	33

6.6 Query Analysis

Previously we showed that on average the quality-based model can effectively improve precision and MRR with respect to the baseline. To understand the limitations of this approach and potentially find improvements, we analyzed the queries for which the model did most poorly in terms of MAP and MRR. Below are the details of six of these queries and our explanations why the quality model does not perform well. Note that in the cases where no MRR is listed, the two models have the same MRR value.

Example 1 (GOV2):

Query	Topic	Baseline	Quality model
Nuclear reactor types	748	0.1138 (MAP) 1.0 (MRR)	0.0628 (MAP) 0.5 (MRR)

Explanation: According to the narrative for this topic, relevant documents only need to mention the names of the types of nuclear reactor power plants. Therefore, low quality documents like lists or tables could be relevant for this topic. The quality model penalizes some of these relevant documents.

Example 2 (GOV2):

Query	Topic	Baseline	Quality model
Green party political views	704	0.1733 (MAP)	0.0662 (MAP)

Explanation: it seems that low quality documents are not likely to be relevant for this topic. However, the narrative section of this topic says "Any members' names noted are considered relevant". There are a few low quality documents judged as relevant only because the names of green party members are listed, which leads to the failure of our model in this case.

Example 3 (WT10G):

Query	Topic	Baseline	Quality model
History of skateboarding	506	0.1276 (MAP) 0.25 (MRR)	0.017 (MAP) 0.026 (MRR)

Explanation: There are only two documents judged as relevant for this topic. One of the two is retrieved by neither the quality model nor the baseline. The other one that is highly ranked by the baseline is a list.

Example 4 (WT10G):

Query	Topic	Baseline	Quality model
Instruments to forecast the weather	541	0.2588 (MAP)	0.1497 (MAP)

Explanation: As in example 1, low quality documents such as lists can be relevant documents for this topic

Example 5 (WT2G):

Query	Topic	Baseline	Quality model
Cuba sugar exports	414	0.5898 (MAP) 1.0 (MRR)	0.4806 (MAP) 0.5 (MRR)

Explanation: In the description section of this topic it says “How much sugar does Cuba export and which countries import it”. As we can see, just numbers and names are enough to be relevant for this topic.

Example 6 (WT2G):

Query	Topic	Baseline	Quality model
Quilts, income	418	0.3643 (MAP)	0.2634 (MAP)

Explanation: The narrative section of this topic states “Documents mentioning quilting books, quilting classes, quilted objects and museum exhibits of quilts are all relevant”. According to these criteria, low quality documents can be relevant for this topic.

In summary, it seems that the biggest problem for the current quality model is that there are queries with relevant documents that are low quality according to the model. To better understand this issue, we manually divided all queries for the Terabyte Track 2004 into the following two types:

Type one: queries that are not likely to have relevant low quality documents.

Type two: queries that are likely to have relevant low quality documents.

According to our classification, there are 33 type one queries and 16 type two queries. The heuristic we used for the classification is that if a few named entities are enough to satisfy the information need as defined in the narrative, the query will be classified as type two. Otherwise, if detailed topic description is needed, the query will be classified as type one. Of course, the classification is still somewhat ambiguous for some queries.

Table 6.8 shows MAP results for the two types of queries defined above. Percentage improvements with respect to the baseline are also given. Considering the explanations given above, it is not surprising to see that the performance of the document quality model is quite low on the type two queries. On the other hand, our model is better than the baseline on the type one queries, although the improvement is small. This is because there are relatively few low quality documents in a typical ranked list and MAP is based on the whole ranked list. Another interesting observation from table 8.1 is that both two models perform better on type one queries. Even though we currently can not automatically distinguish the two types of queries, our analysis suggests that a different strategy is needed to improve the performance of type two queries.

Table 6.8 : MAP on the two types of queries

Query Type	Query likelihood model	Document quality model
Type One	0.2664	0.2710 (+1.7%)
Type Two	0.2209	0.2045 (-7.4%)

CHAPTER 7

CONCLUSION AND FUTURE WORK

7.1 Conclusions

The contributions of this thesis have been stated in the introduction chapter of this thesis. We summarize them as follows:

- Defining the area of performance prediction
- Describing a comprehensive range of experiments for performance prediction
- Showing that satisfactory prediction accuracy is achievable across a variety of search scenarios
- Demonstrating the superiority of WIG in Web search environments
- A framework for query expansion prediction
- A document quality language model incorporating quality features for Web retrieval

In addition, major insights and lessons gained from this work are:

(1) Collection types and query types have a significant impact on the accuracy of a prediction technique.

(2) As is the case for retrieval, term proximity can be helpful for performance prediction, although carefully modeling term proximity is required.

(3) Performance prediction techniques designed for content-based queries generally do not perform well for named-page finding queries.

(4) Regarding performance prediction, there exists a uniformed framework for dealing with performance prediction for both content-based and named-page finding queries.

(5) In general, the performance difference between two retrieval techniques for a given query is smaller than the difference between two queries for a given retrieval technique. Therefore, query expansion prediction is a harder problem than query performance prediction.

(6) The notion of quality within the context of IR is necessary deserves further investigation.

7.2 Future Work

There are many interesting extensions to this thesis work. Here we highlight some of them.

(1) *Improving Prediction Models*

Our most promising performance prediction model is WIG, but one issue with WIG is its sensitivity to a description query. One naïve way would be automatically compressing the description query into a few key words. Since the compression process is not perfect, we would like to see if this strategy can eventually improve the prediction accuracy of WIG for description queries. Additionally, the query feedback method leaves much room for improvement. For example, exploring alternative ways of implementing the rank list distillation part has the potential of making this technique computationally efficient.

(2) *Ranking Robustness and Ranking Functions*

We have shown that the ranking robustness technique is capable of predicting query performance. In that case, the ranking function is fixed and we compare robustness scores across a set of queries. Another interesting question is: given a set of queries and more than one ranking functions, can we utilize the idea of ranking robustness to select the best ranking function for that particular set of queries? For example, as a first step, we can compare the robustness scores of two kinds of ranking functions: document likelihood and query likelihood. Previous studies have shown empirically that using query likelihood leads to much better retrieval performance compared to document likelihood. However, there is no theoretical justification for choosing one over the other. It would be interesting to investigate whether the superior performance of query likelihood can be explained under our ranking robustness framework.

(3) *Applications in Distributed IR*

Retrieval performance can be viewed as a function of query Q and collection C. In this thesis, we focus on the situation when collection C is fixed but query Q is a variable. We are also interested in the reverse situation: query Q is fixed but collection C is a variable. This situation corresponds to the collection selection problem in distributed IR. With the help of a performance predictor, those collections with the highest predicted performance for a given query will be chosen. We believe that taking performance prediction into account has the potential of significantly improving selection accuracy. Moreover, performance predication can be applied to the problem of merging document rankings. Traditional methods for results merging are based on the normalization of document scores from different search engines. We can incorporate performance prediction by assigning a weight learned from a performance predictor to each of these search engines. Search results are merged based on these performance weights associated to the search engines.

(4) *Query Disambiguation*

Retrieval effectiveness suffers greatly from users' ambiguous queries, notably in Web search where most users tend to submit very short queries with little context. On the other hand, formulating a well-defined information request is still challenging even for experts. The performance prediction techniques developed in this thesis can play an important role in query disambiguation by either incorporating them into existing methods or by providing insights to develop new disambiguation techniques. For example, one way used by many commercial search engines for query disambiguation is the technique called query refinement that allows the user to interactively specify her information need by selecting new terms suggested by the system. Prediction techniques will provide guidance on important issues such as what kind of queries need refinement and how to select effective terms based on the user's original query.

(5) *Information Quality*

Information quality has become a significant concern especially on the Web where little restriction is placed on generating and publishing web documents. Relevant but fraudulent information is much worse than irrelevant but reliable information. Modeling information quality is difficult because it is highly user-dependant and involves many aspects such as authority, accuracy, objectivity and timeliness. The document quality chapter in this thesis only addresses one aspect of information quality and the features used are solely based on term statistics. We would like to explore more features related to quality, especially user-interaction features such as click information. We also plan to further investigate the relationship between quality and relevance, and potentially develop different quality measures for different types of queries.

APPENDIX A : RETRIEVAL PERFORMANCE MEASURES

In this thesis, we use the following three measures for evaluating the performance of the retrieval results in response to a given query: precision at a given cut-off level, Mean Reciprocal Rank (MRR) and average precision. Specifically, we assume that the retrieval results are in the form of a ranked list of documents, that is , $\{d_1, d_2, ..d_n\}$ where n is the length of the ranked list and d_i is the i-th ranked document. We also assume the relevance judgments is in the form of a binary vector $R = \{r_1, r_2, ...r_n\}$ where $r_i = 1$ if document d_i is relevant or $r_i = 0$ if document d_i is irrelevant. Next we give details on these measures.

(1) Precision at a given cut-off rank k: $precision(k)$

It is defined as follows:

$$precision(k) = \frac{\sum_{i=1}^{k} r_i}{k}, k \le n$$

That is, precision at cut-off rank k is the percentage of relevant documents in the top k retrieved documents. This measure is often used for ad-hoc retrieval evaluation in a Web search environment where a user generally looks at no more than the first one or two pages of results.

(2) Mean Reciprocal Rank : MRR

MRR is defined as the inverse of the rank of the first relevant document, that is,

$$MRR = \frac{1}{j}, where \sum_{i=1}^{j-1} r_i = 0 \; and \; r_j = 1$$

MRR is useful in cases where users are primarily looking for one correct answer and want that answer ranked as high as possible. Named-Page finding and question answering are two TREC tasks where MRR is the standard for evaluation.

(3) Average Precision

Average precision is frequently used for ad-hoc retrieval. It emphasized returning more relevance document earlier. It is the arithmetic mean of precisions calculated at each of the relevant documents. Mathematically speaking,

$$average\ precision = \frac{\sum_{i=1}^{n} precision(i) \times r_i}{\sum_{i=1}^{n} r_i}$$

BIBLIOGRAPHY

[1] Voorhees,E.M. Overview of the TREC 2004 Robust Track. In *the Online Proceedings of 2004 Text REtrieval Conference* (TREC 2004).

[2] Robust Track   http://trec.nist.gov/tracks.html

[3] Harman, D and Buckley,C. The NRRC Reliable Information Acess(RIA) workshop. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval.(SIGIR 2004)* ,pp. 528-529.

[4] Zhai,C and Lafferty,J. A study of smoothing methods for language models applied to Ad Hoc information retrieval. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval.(SIGIR 2001) ,* pp.334-342.

[5] Buckley,C. Why current IR engines fail. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval.(SIGIR 2004),* pp.584-585.

[6] Yom-Tov,E., Fine,S., Carmel,D., Darlow,A.   Learning to Estimate Query Difficulty with Applications to Missing Content Detection and Distributed Information Retrieval. In *28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2005),* pp. 512-219.

[7] Zhou,Y. , Croft,W.B., Ranking Robustness: A Novel Framework to Predict Query Performance, in *Proceedings of the 15th ACM international conference on Information and knowledge management (CIKM 2006),* pp 546-574.

[8] Carmel,D., Yom-Tov,E., Darlow,A.,Pelleg,D. What Makes a Query Difficult? in *29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2006)*,pp.390-397.

[9] Clarke,C.L.A., Scholer,F., Soboroff,I. The TREC 2005 Terabyte Track, In *the Online Proceedings of 2005 TREC.*

[10] Text REtrieval Conference (TREC) Home Page, http://trec.nist.gov/

[11] Croft,W.B. , Lafferty,J. Language Models for Information Retrieval. in *Kluwer Academic Publishers*, 2003.

[12] Brin,S., Page,L., The anatomy of a large-scale hypertextual web search engine. In *Computer Network and ISDN Systems*, 30(1-7):107-117,1998.

[13] Kleinberg,J.M. Authoritative sources in a hyperlinked environment. In *Journal of the ACM*, 46(5):604-632,1999.

[14] Westerveld,T., Kraaij,W., Hiemstra,D. Retrieving web pages using contents, links, URL's and anchors. In *the Tenth Text Retrieval Conferences (TREC-2001)*, pp. 52-61.

[15] Wessel,K., Thijs,W., Djoerd, H. The importance of prior probabilities for entry page search. in *25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2002)*, pp. 27-34.

[16] Hawking,D. Overview of the TREC-9 web track. In *the Ninth Text Retrieval Conferences (TREC9)*, pp. 87-102.

[17] Hawking,D.,Voorhees,E.,Craswell,N., Bailey,P. Overview of the TREC-8 web track. In *the Eight Text Retrieval Conference (TREC8)*,pp.131-148.

[18] Hawking,D., Craswell,N. Overview of the TREC-2001 web track. In *The Tenth Text Retrieval Conferences (TREC-2001)*, pp. 25-31.

[19] Cronen-Townsend,S., Zhou,Y., Croft,W.B. Predicting query performance. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval.(SIGIR 2002)* , pp. 299-306.

[20] Amati,G., Carpineto,C., Romano,G. Query difficulty, robustness and selective application of query expansion In *European Conference on Information (ECIR 2004)* ,pp 127-137

[21] Predicting Query Difficulty. SIGIR workshop 2005 http://www.haifa.ibm.com/sigir05-qp/index.html

[22] Tomlinson,S., Robust,Web and Terabyte Retrieval with Hummingbird SearchServer at TREC 2004. In *the Online Proceedings of 2004 Text REtrieval Conference (TREC 2004)*

[23] He,B.,Ounis,I. Inferring query performance using pre-retrieval predictors. In *proceedings of the SPIRE 2004*. pp 43-54.

[24] Plachouras,V., He,B., Ounis,I. University of Glasgow at TREC 2004: Experiments in Web, Robust, and Terabyte Tracks with Terrier. In *the Online Proceedings of 2004 Text REtrieval Conference (TREC 2004)*

[25] Diaz,F., Jones,R. Using temporal profiles of queries for precision prediction. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval.(SIGIR 2004)*, pp. 18-24.

[26] Kwok,K.L., Grunfeld,L., Sun,H.L., Deng,P. TREC 2004 Robust Track Experiments Using PIRCS. In *the Online Proceedings of 2004 Text REtrieval Conference (TREC 2004)*

[27] Bernstein, Y., Billerbeck,B., Garcia,S., Lester,N., Scholer,F., Zobe,J. RMIT University at TREC 2005: Terabyte and Robust Track. In *the Online Proceedings of 2005 Text REtrieval Conference (TREC 2005)*

[28] Jensen,E.C., Beitzel,S.M.,Chowdhury,A., Frieder,O., Grossman,D. Predicting Query Difficulty on the Web by Learning Visual Clues. In *Proceedings of the 2005 ACM Conference on Research and Development in Information Retrieval (SIGIR 2005)*, pp. 615-616

[29] Vinay,V., Cox,I.J.,Mill-Frayling,N.,Wood,K. On Ranking the Effectiveness of Search, in *Proceedings of the 2006 ACM Conference on Research and Development in Information Retrieval (SIGIR 2006)*, pp. 398-404.

[30] Personal email contact with Vishwa Vinay and our own experiments

[31] Kwok,K.L., Grunfeld,L., Dinstl,N., Deng,P. TREC 2005 Robust Track Experiments Using PIRCS. In *the Online Proceedings of 2005 Text REtrieval Conference (TREC 2005)*

[32] Robertson,S. 1984, On term selection for query expansion, in *Journal of Documentation* Vol. 46, pp.359-364

[33] Xu,J., Croft,W.B ,2000, Improving the effectiveness of information retrieval with local context analysis. In *ACM Transaction on Information Systems* 18(1), pp. 79-112.

[34] Yang,K. Combing text and link-based retrieval method for web IR. In *the Tenth Text Retrieval Conferences (TREC-2001)*, pp.609-618.

[35] Kraaij,W., Westerveld,T. TNO-UT at TREC-9: How Different are Web Documents? In *the Ninth Text Retrieval Conferences (TREC9),* pp. 665-672.

[36] Hiemstra,D., Kraaij,W. Twenty-one at TREC-7: Ad-hoc and cross-language track. In *the Seventh Text Retrieval Conferences (TREC7*), pp. 334-342.

[37] Li,X., Croft,W.B. Time-based language models. In *proceedings of the Twelfth International Conference on Information and Knowledge Management (CIKM 2003),* pp.469-475.

[38] Wessel,K., Thijs,W., Djoerd,H. The importance of prior probabilities for entry page search. In *Proceedings of the 25nd annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR 2002),* pp.27-34.

[39] Zhu,X., Gauch,S. Incorporating quality metrics in centralized/distributed information retrieval on the World Wide Web. In *Proceedings of the 23nd annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR 2000),*pp. 288-295.

[40] Lavrenko, V., Croft, W.B. Relevance-Based Language Models, In *Proceedings of the 24nd annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR 2001),* pp.120-127.

[41] Song,F., Croft,W.B. A general language model for information retrieval. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval. (SIGIR 1999),* pp.279-280.

[42] D.Lopresti and J.Zhou, Retrieval Strategy for Noisy Text, In symposium on document analysis and information retrieval, pp1-16,1996

[43] Singhal,A.,Salton,G., Buckley,C. Length normalization in degraded text collections. In *symposium on document analysis and information retrievl(1996),* pp.149-162.

[44] Mittendorf,E. Data corruption and information retrieval, PhD Thesis, Department of Computer Science, the Katholieke Universiteit Leuven.

[45] Gibbons,J.D, Chakraborty,S. Nonparametric statistical inference. Marcel Dekker, New York,1992

[46]Bookstein,A. ,Swanson,D. Probabilistic models for automatic indexing. In *Journal of the American Society for Information Science* 25,5(1974), pp. 312-319.

[47]Harter,S.P. A probabilistic approach to automatic keyword indexing. Journal of the American Society for Information Science 26,4 and 5(1975), Part I:197-206; Part II:280-289

[48] M.H. Kalos and P.A. Whitlock, Monte carlo methods, John Wiley & Sons,Inc. 1996

[49] Berger,A., Lafferty,J. Information retrieval as statistical translation. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval. (SIGIR 1999)*, pp. 222-229.

[50] Cronen-Townsend,S., Zhou,Y., Croft,W.B. A Framework for Selective Query Expansion, a poster presentation, in *proceedings of the 13th International Conference on Information and Knowledge Management (CIKM 2004)*, pp.236-237

[51]Metzler,D., Croft,W.B. A Markov Random Filed Model for Term Dependencies, in *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval. (SIGIR 2005)*, pp.472-479.

[52]I.J. Taneja: On Generalized Information Measures and Their Applications, In *Advances in Electronics and Electron Physics*, Academic Press (USA), 76, 1989, pp. 327-413.

[53]Metzler,D., Strohman,T.,Zhou,Y.,Croft,W.B. Indri at TREC 2005: Terabyte Track, In *the Online Proceedings of 2004 TREC*.

[54]Ogilvie,P., Callan,J. Combining document representations for known-item search, in *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval. (SIGIR 2003)*, pp. 143-150.

[55] Kreyszig,E. Advanced Engineer Mathematics, John Wiley & Sons,Inc. 1997

[56] TREC Terabyte Track http://www-nlpir.nist.gov/projects/terabyte/

[57] Web Research Collections http://ir.dcs.gla.ac.uk/test_collections/

[58] N.Jardine and C.J.V.Rijsbergen. The use of hierarchic clustering in information retrieval. In *Information Storage and Retrieval*, Vol 7,pp.217-240,1971

[59] T. Hastie, R. Tibshirani, J. H. Friedman .The Elements of Statistical Learning, Springer press, 2001

[60] Zhou,Y., Croft,W.B. Document quality models for Web ad-hoc retrieval, a poster presentation, in *proceedings of the 14th International Conference on Information and Knowledge Management (CIKM 2005)*, pp. 331-334.

[61] George Casella and Roger L. Berger, Statistical Inference, Duxbury 2002.

[62] The Indri search engine http://www.lemurproject.org/indri/

[63] G. Kumaran and J. Allan, Selective User Interaction. To appear in *the Proceedings of the ACM Sixteenth Conference on Information and Knowledge Management (CIKM 2007).*

[64] A. Broder. A taxonomy of web search. SIGIR Forum, 36(2),2002.

[65] Beitzel,S.M., Jensen,E.C., Frieder,O. Automatic web query classification using labeled and unlabeled training data, In *the Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*(SIGIR 2005) pp. 581-582.

[66] In-Ho Kang, GilChang Kim, Query type classification for web document retrieval, In *the Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR 2003)* pp. 64-71.

[67] Diaz, F. Performance Prediction Using Spatial Autocorrelation. In *the Proceedings of the 30th Annual International ACM SIGIR Conference (SIGIR 2007)*, pp. 583-590.

[68] Zhou,Y, Croft. W.B. Query performance prediction in Web search environments . in the Proceedings of the 30th Annual International ACM SIGIR Conference (2007), pp. 543-550.