Optimizing the Construction of Information Retrieval Test Collections

Mehdi Hosseini

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

of the

University College London.

Department of Computer Science

University College London

January 13, 2013

Declaration

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

Thesis Committee

Primary Supervisor: Prof. Ingemar J.Cox, University College London

Industrial Supervisor: Dr. Natasa Milic-Frayling, Microsoft Research

Secondary Supervisor: Dr. Jun Wang, University College London

Internal Examiner: Dr. Licia Capra, University College London

External Examiner: Dr. Leif Azzopardi, University of Glasgow

Abstract

We consider the problem of optimally allocating a limited budget to acquire relevance judgments when constructing an information retrieval test collection. We assume that there is a large set of test queries, for each of which a large number of documents need to be judged. However, the available budget only permits to judge a subset of them.

We begin by developing a mathematical framework for query selection as a mechanism for reducing the cost of constructing information retrieval test collections. The mathematical framework provides valuable insights into properties of the optimal subset of queries. These are that the optimal subset of queries should be least correlated with one another, but have a strong correlation with the rest of queries. In contrast to previous work, which is mostly retrospective, our mathematical framework does not assume that relevance judgments are available a priori, and hence is designed to work in practice.

The mathematical framework is then extended to accommodate both the query selection and document selection approaches to arrive at a unified budget allocation method that prioritizes query-document pairs and selects a subset of them with the highest priority scores to be judged. The unified budget allocation is formulated as a convex optimization, thereby permitting efficient solution and providing a flexible framework to incorporate various optimization constraints.

Once a subset of query-document pairs are selected, crowdsourcing can be used to collect associated relevance judgments. While the labels provided by crowdsourcing are relatively inexpensive, they vary in quality, introducing noise into the relevance judgments. To deal with noisy relevance judgments, multiple labels for a document are collected from different assessors. It is common practice in information retrieval to use majority voting to aggregate multiple labels. In contrast, we develop a probabilistic model that provides accurate relevance judgments with a smaller number of labels collected per document.

We demonstrate the effectiveness of our cost optimization approach on three experimental data, namely: (*i*) various TREC tracks, (*ii*) a web test collection of an online search engine, and (*iii*) crowdsourced data collected for the INEX 2010 Book Search track.

Our approach should assist research institutes, e.g. National Institute and Standard Technology (NIST), and commercial search engines, e.g. Google and Bing, to construct test collections where there are large document collections and large query logs, but where economic constraints prohibit gathering comprehensive relevance judgments.

Acknowledgements

Many people have helped me come to this place in my education. Foremost among them is my supervisor, Ingemar Cox, who offered me a full scholarship and led me to come to UCL. Because of Ingemar's confidence in me I was able to explore research directions with great freedom while he was always supporting me at critical times with new suggestions, insights and honest feedback. I am fortunate to have had Ingemar as my supervisor and have learned a much from him.

I owe a great deal to Microsoft Research for sponsoring my PhD, and in particular my industrial supervisor, Natasa Milic-Frayling, who fully supported me throughout the PhD. Also, much of this work was done in collaboration with my friends in Microsoft, including Vishwa Vinay, Milad Shokouhi, Gabriella Kazai and Emine Yilmaz. Vinay's keen insights, feedback, and encouragement are much appreciated. Conversations with him always helped me find the directions toward the end of my PhD.

I am honored to have the privilege to work with Trevor Sweeting who so kindly opened my eyes to the beauty of mathematical modeling and optimization. I also thank Stephen Robertson, Jun Wang and Jianhan Zhu for early discussions on my work.

My parents and sisters have been endlessly supportive and patient, even when I failed to call for long times. They have always been willing to come to my side when I needed them, and for that I thank them deeply. I hope that I can provide the same when they need me.

Finally, none of this work would have been possible without the unwavering love and support of my sweet Mandana, who spent many long days and sleepless nights, sometimes alone, while I was working on this thesis. I dedicate this thesis to her who is the most important and inspiring outcome from my time in UK.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In information retrieval (IR) experiments an Ad-hoc test collection is used to evaluate the performance of various retrieval systems. An Ad-hoc test collection consists of (*i*) a corpus of documents, (*ii*) a set of queries (topics in TREC terminology), and (*iii*) a set of relevance judgments for each query. Relevance judgments indicate which documents in the corpus are relevant to a particular query. In a typical evaluation of a retrieval system, an effectiveness metric, e.g. Average Precision, receives associated relevance judgments as input and measures the system's performance for a query. The system's average performance is then measured based on its performance scores measured across a set of queries. Finally, systems are ranked based on their average performance.

1.1 Low-Cost Test Collections

Starting from early Ad-hoc test collections, with a few thousands of documents and tens of queries, information retrieval experiments have transitioned to test collections with billions of documents, and aspire to systems evaluation over millions of queries. However, despite this increase in size, it still remains necessary to manually acquire relevance judgments in order to calculate retrieval effectiveness metrics. When the corpus and the number of queries were small, it was feasible to acquire relevance judgments by employing a number of human assessors who compared every document or at least a sufficiently large number of documents in the corpus to every query. However, when the corpus and the number of queries are large, this is no longer feasible, due to both the economic cost and time involved.

The cost of gathering relevance judgments, in its simplest form, is a function of the number of queries chosen to evaluate the retrieval systems, the number of documents judged per query, and the human effort spent on judging a document. However, cost is not the only criterion. Reliability and accuracy of conclusions drawn by using a test collection are also extremely important. Indeed, a robust test collection has no inherent bias that might affect the evaluation of any retrieval systems.

In a common scenario of IR experiments, a large set of queries are initially compiled against which we desire to measure the performance of a set of systems. Ideally, a system can be evaluated and compared with other systems if we manually assess a significant portion of the document corpus or, at least, a large number of documents retrieved by each individual system.[1] However, the available budget

[1] This is the case when recall sensitive metrics like average precision and Recall are used to measure a system's performance.

prohibits gathering exhaustive relevance judgments by expert assessors. We can efficiently deal with the budget constraint by:

- minimizing the number of queries for which relevance judgments are required. This approach is known as *query selection* and motivated by the retrospective experiments conducted by Guiver et al. [GMR09], showing that it is possible to reproduce the results of an exhaustive evaluation of systems over many queries by using a much reduced set of queries that is representative of the full set.

- minimizing the number of documents that need to be judged for a query. This approach is referred to as *document selection*. There exists a rich body of related work on designing efficient document selection methods, e.g. Carterette et al. [CAS06] and Aslam et al. [APY06], supported by metrics designed for shallow relevance judgments, e.g. Carterette et al. [CAS06] and Yilmaz et al. [YA06].

- outsourcing the assessment task to a large group of assessors via crowdsourcing experiments instead of assigning the task to a few experts. Web services like Amazon Mechanical Turk[2] provide facilities to temporarily hire a large number of crowd assessors to collect relevance judgments with a minimum cost and in a short period of time. While still in the early stages, the practices of outsourcing the relevance judgment tasks are evolving and the IR community is investigating the benefits and the drawbacks of the crowdsourcing approach [Alo11].

Despite a large amount of study on the document selection problem, little literature is available on the query selection problem. The characteristics that a subset of queries should hold to be representative of the full set of queries is still unclear. Also, previous work did not address how the number of queries can be effectively minimized when relevance labels are not available yet. In addition, an approach that combines the various aspects of query selection and document selection and provides a unified optimization framework has not been addressed yet. Finally, efficiently dealing with the noise in relevance labels, provided by crowd assessors, is still one of the main challenges in crowdsourcing experiments.

1.2 Contributions

This work expands the existing research in three directions: (*i*) formulating the query selection problem and designing a solution model that effectively performs in practice, (*ii*) modeling a unified optimization framework that combines various aspects of document selection and query selection and provides criteria for constructing robust test collections under the budget constraint, and (*iii*) modeling the noise in crowdsourcing experiments to efficiently aggregate multiple noisy labels and construct relevance judgments.

This work will be of significant benefit in constructing low-cost test collections for the IR community or the commercial search engines like Bing and Google, where there are very large document collections and query logs, but budget constraints prohibit providing the comprehensive set of relevance judgments.

[2] https://www.mturk.com/

1.2.1 Query Selection

The query selection problem is essentially important when a large set of queries is initially compiled against which we desire to measure the performance of a set of systems. However the available budget only permits relevance judgments for a subset of queries. The goal of a query selection method is to find a subset of queries that most closely approximates evaluation results that would be obtained if one used the full set of queries. The common approach for selecting a subset of queries is random sampling. However, Guiver et al. [GMR09] recently showed that query subsets vary in their accuracy of predicting the systems' average performance that is computed over the full set of queries. Their results indicate that particular subsets of queries, known as *representative* subsets, are good predictors of the systems' average performance. However, it is still unclear what properties the representative queries should contain, and how to select such a representative subset when relevance judgments are not available yet.

We first assume relevance judgments are available for all the queries in a test collection, and develop a mathematical framework for query selection (Chapter 4). The mathematical framework provides valuable insights into the characteristics of the optimal subset of queries. These are that the optimal subset of queries (*i*) are least correlated with one another, thereby maximizing the information we gain from each, and (*ii*) should have strong correlation with the remaining queries, as without this correlation there is no predictive capability.

We then relax the assumption that relevance judgments are available before selecting queries and extend the mathematical framework (Chapter 6). In particular, our mathematical framework explicitly models uncertainty in the retrieval effectiveness metrics that are introduced by the absence of relevance judgments. Thus, in contrast to previous work which is retrospective and assumes some relevant judgments are available for each query, e.g. Guiver et al. [GMR09] and Robertson [Rob11], our approach is designed to work in practice and does not require the existence of prior relevance judgments.

Since the optimization model is computationally intractable, we devise an adaptive query selection algorithm to provide an approximate solution. We demonstrate the effectiveness of the adaptive algorithm using various test collections including a large scale dataset of a commercial search engine. The experimental results prove that the adaptive method could reduce at least 35% of queries that are required by the considered baseline methods to obtain 90% accuracy in ranking the retrieval systems. We also investigate how the selected query subset generalizes to (*i*) new unseen systems and (*ii*) changes to the evaluation metric. We show that the adaptive algorithm can be modified to improve generalizability in both cases.

1.2.2 Reusability

Recent studies have concentrated on IR evaluations with large query sets, e.g. Carterette et al. [CPK$^+$08], aided by document selection methods that reduce the number of relevance judgments per query in order to make relevance judgments feasible, e.g. Carterette et al. [CAS06], as well as introducing evaluation metrics for partially judged result sets, e.g. Yilmaz et al. [YA06]. However, due to a small number of documents assessed per query, the reusability of such a test collection still remains questionable [CKPF10].

New as yet unseen systems may return many documents that are previously unjudged. Thus, relying on the current set of relevance judgments may cause high uncertainty in measuring the performance of the new systems.

We show how a query selection approach can be used to maintain a test collection reusable (Chapter 5). We assume a fixed budget to build extra relevance judgments over documents that are solely retrieved by the new systems. We extend our query selection framework to arrive at a budget-constrained optimization. We use the budget-constrained optimization to select a representative subset of queries, and allocate the budget to build relevance judgments only for the selected queries. We then estimate the new systems' average performance based on the selected queries. Our experimental results show that spending the fixed budget on the subset of queries produces more accurate estimates of the average performance of the new systems than spreading the budget uniformly across all the queries.

Such a scenario is particularly useful when the evaluation is being conducted by small groups of researchers investigating their new retrieval systems by large scale test collections, e.g. TREC Million Query track [ACA+07]. However, the initial set of relevance judgments may be insufficient to reliably evaluate the new systems, and there is a limited budget to construct relevance judgments for a subset of previously unjudged documents.

1.2.3 Unified Budget Allocation

The mathematical framework is extended to combine the query selection and the document selection approaches to arrive at a unified optimization framework that selectively chooses a subset of query-document pairs to build relevance judgment under a budget constraint (Chapter 7). The optimization framework first assigns a priority score to each of the queries. Next, a set of documents retrieved in response to a query are prioritized by using an efficient document selection method, e.g. Carterette et al. [AP08]. The queries and documents are then combined to form the priority scores for the query-document pairs. Finally a subset of query-document pairs with the highest priority scores are selected to collect relevance judgments. We evaluate our budget allocation approach using various TREC test collections. We demonstrate that our budget allocation is cost efficient and yields a significant improvement over the considered baselines.

1.2.4 Crowdsourcing

Crowdsourcing experiments are used to collect relevance judgments by temporarily hiring a large number of crowd assessors. While the labels provided by the assessors are relatively inexpensive, they vary in quality, introducing noise into the relevance judgments, and consequently causing inaccuracies in the system evaluation [KKKMF11]. In order to address the issue of noisy labels, it is common to collect multiple labels from different assessors, in the hope that the consensus across multiple labels would lead to more accurate relevance judgments. Common practices in information retrieval use the majority voting to aggregate multiple labels and infer relevance of a document. Using the majority voting, sometimes a large number of labels are required to truly predict the relevance of a document [AM09]. However, if the number of labels required for a document is large, no benefit in cost is achieved against the traditional methods that collect only one expensive label per document from an expert assessor.

We develop a probabilistic model as an alternative to majority voting that needs fewer labels to correctly infer a document's relevance (Chapter 8). In contrast to previous work, we assume that no authoritative relevance judgements are available, to be used as training data, and estimate both the accuracy of the assessors and the document relevance from the noisy labels.

We run simulations and conduct experiments with crowdsourced data to investigate the accuracy and robustness of the relevance judgments to the noisy labels. Our experimental results show that the probabilistic model outperforms the majority voting method in terms of both the accuracy of relevance assessments and the ranking of IR systems.

1.2.5 Example

To better understand how the techniques developed in this thesis can be used in practice consider a search engine company that is seeking the best performing retrieval model among a set of candidate retrieval models to be used in its commercial search engine. The company creates a large scale test collection with a large number of queries, extracted from the search engine's query log, and a large number of documents.

To measure the performance of retrieval models, the company has to hire a set of human assessors to create relevance judgments. However, gathering relevance judgments for all the queries and documents in the test collection is prohibitively expensive and time consuming. The company could use the document selection techniques like the pooling mechanism or a recently developed document selection method, e.g. Carterette et al. [CAS06] and Aslam et al. [APY06], to decrease the number of documents judged per query, and hence to reduce the cost of evaluation. However, since there are a large number of queries, still a large amount of budget is required to gather relevance judgments.

The company can alternatively use the unified budget allocation method developed in this thesis which not only makes use of the document selection methods to reduce the number of documents judged per query but also develops a query selection mechanism to minimize the number of queries used to create relevance judgments. Hence, it minimizes the overall cost of creating a test collection by minimizing the number of queries and documents.

1.3 Organization

This thesis is organized as bellow:

- **Chapter 2 — Background**: includes an overview of information retrieval and introduces terminology that will be used in the rest of the thesis.
- **Chapter 3 — Cost Effective IR Test Collections**: represents the history of developments in low-cost IR test collections, and defines the notion of cost and introduces related work.
- **Chapter 4 — The Query Selection Problem**: defines the query selection problem in details. It is assumed that relevance judgments are available for the full set of queries. Query selection is then formulated as an optimization problem. Finally, the characteristics of the optimal subset of queries are investigated.

- **Chapter 5 — The Reusability of a Test Collection**: defines the concept of reusability of a test collection. It is shown how a query selection approach can be used to enhance the reusability of a test collection under a budget constraint.
- **Chapter 6 — Uncertainty-Aware Query Selection**: relaxes the assumption made in Chapter 4 that relevance judgments are available before selecting queries. The mathematical framework, formulating the query selection problem, is extended to model uncertainty that is due to absence of relevance judgments. An adaptive query selection algorithm is then proposed to implement the theoretical framework in practice.
- **Chapter 7 — Unified Budget Allocation**: a unified cost-optimization is proposed to combine the query selection and document selection approach to arrive at a query-document pair selection approach.
- **Chapter 8 — Crowdsourcing Relevance Judgments**: a probabilistic approach is proposed to model the noise in relevance labels collected by crowdsourcing experiments, and integrate multiple noisy labels to construct relevance judgments.
- **Chapter 9 — Conclusion**

Chapter 2

Background

The basic task of information retrieval (IR) is to find relevant information in response to a user need. Consequently, IR systems are concerned with issues such as collecting, indexing, searching and displaying information. A well-known example of an IR system is a web search engine.

The broad definition of IR above will be further discussed in detail. However, before doing so, it is useful to define some basic concepts and terminology.

Ad-hoc retrieval is the most standard task in IR and refers to the retrieval of files that match a user's need. The form of the files usually is *text documents*. The text of an email or a single web-page is an example of a text document. Alternatively, the files could be images, audio, or videos. A text document is composed of a set of terms where a term means:

- A word in the document.
- A stemmed word showing the stem of words with the same root but different forms.
- A phrase that is a group of words that together have a particular meaning, e.g. *information retrieval*.

The group of documents, to be searched, is called a *collection*. A collection could, for example, be a personal collection of publications, emails, or a large collection of web-pages. Each document in the collection is considered as *relevant* if it satisfies the user's information need. Relevance is a subjective concept which is highly dependent to users judgment. It is considered either a binary variable (relevant or non-relevant) or a continuous variable showing the degree of relevance. The user's need is formulated by a *query* (topic) consisting of a set of words. Documents and the user query are the inputs to a *retrieval model*. The retrieval model is the main component of ad-hoc retrieval and its main task is to assign a score to each document indicating its potential relevance to the query.

There are alternatives to the ad-hoc retrieval task. For example, a *question answering* (*QA*) system provides an answer in response to a user's need. The user's need is usually expressed by some question sentences, and *QA* provides an answer to each question in turn. The form of answers is varied. Depending on the question, sometimes the answer is only one sentence and sometimes it is expressed in several paragraphs. As a another example, a Cross-language retrieval system finds documents that pertain to a query regardless of the language in which the document is written.

	d_1	d_2	d_3	d_4	d_5	d_6	...
t_1	1	0	1	0	1	0	
t_2	1	1	1	0	1	0	
t_3	1	0	0	0	1	1	
t_4	1	0	0	1	0	0	
...							

Figure 2.1: A term-document incidence matrix.

2.1 The Main Components of an Ad-hoc Retrieval Task

The key components of an ad-hoc retrieval are:

- representation
- retrieval model
- result set
- relevance feedback

2.1.1 Representation

The documents and query have to be represented in a comparable form so that the retrieval model is able to compare each document to the query and estimate its relevance. The type of representation depends on the retrieval model being used. The conversion of the query to the representation model occurs during search time and just after being entered by the user, but documents are typically converted in advance.

A document is represented as a vector of terms. Each element of the vector represents a unique term and is assigned a weight. There are several ways to define the weights. A weight can be a binary variable showing absent or present of the corresponding term in the document, or it can be a continuous variable measured by a *weighting function*. The weighting function assigns a value to each term which corresponds to the degree to which the term characterizes the document. The term frequency-inverse document frequency (*tf-idf*) is a well-known weighting function consisting of two parts: the fist part is term frequency (*tf*) that denotes how often a term occurs in a document, and the second part is inverse document frequency (*idf*). The *idf* of a term is the inverse of the fraction of documents in the collection that contain this term.

If we consider each document as a vector of terms, the collection can itself be represented as a term-document matrix in which each row refers to a term and each column refers to a document. The *binary term-document incidence* matrix is a representation in which the matrix element (t, d) is 1 if the document in column d contains the term in row t, and is 0 otherwise. Figure 2.1 shows an example of the binary term-document matrix.

The collection usually contains a large number of documents, and the number of documents that contain a term is usually small. As a result, the corresponding matrix is extremely large and sparse, and representing all the cells that contain zero is inefficient. A more efficient method for representing the matrix is recording terms in a linked list. Each item in the linked list, referred to as a posting, points to

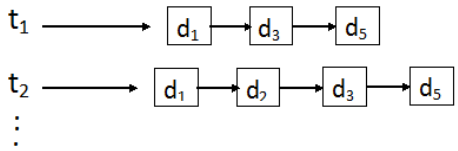


Figure 2.2: An inverted index for a part of the matrix in Figure 2.1.

documents containing the term. This approach, called an *inverted index*, requires less space to store the data. Figure 2.2 shows a part of the matrix in Figure 2.1 represented as an inverted index.

A user's query is also represented as a vector of terms. After a user submits a query, it is converted to the representation form, and sent to a retrieval model. In addition to the query, the retrieval model accepts documents as the other input parameter. In the next section we introduce several retrieval models.

2.1.2 Retrieval Models

A retrieval model compares each document in the collection to the given query. Retrieval models are divided in two categories. The first group of retrieval models only separate relevant from non-relevant documents, whereas the second group measures the degree of relevance for each document. The models in the second group have a scoring function that estimates each document's relevance. In the following some standard retrieval models are discussed in detail.

- ***Boolean Model***: The Boolean model separates relevant documents from non-relevant ones. The query is a Boolean expression. A Boolean expression is a combination of terms with Boolean operators, e.g. AND, OR, and NOT. Initially, documents and the given query are represented as vectors with binary values showing absence or present of the terms. In the next step, the Boolean model runs a series of Boolean operations, and finally separates relevant from non-relevant documents via a binary string. A binary string is a sequence of bits. Each bit is either 0 or 1. If a bit is 1, it means the corresponding document is relevant. For example, in Figure 2.1, the query t_1 AND t_2 AND NOT t_4 is satisfied by the third and fifth documents, D_3 and D_5, since they contain both the terms t_1 and t_2 and exclude t_4. In order to find the relevant documents, the Boolean model takes the rows of t_1, t_2 and t_4, complements t_4, and then does a bitwise AND:

  101010 AND 111010 AND 011011 = 001010

- ***Vector Space Model***: In this model a document is represented as a vector of weights. Each weight indicates the significance of a term in the document measured by a weighting function. The query is represented in the same way. The model estimates the degree of relevance of each document by measuring the similarity between the query and document vectors. The similarity is measured via a scoring function. In ad-hoc retrieval, the commonly used scoring function is the cosine dot product. This measure is the cosine of the angle between the two vectors in the term space. For example, in Figure 2.3 the term space is formed by two terms, t_1 and t_2. The angle between the document vector $\overrightarrow{d}$ and query vector $\overrightarrow{q}$ is θ. The similarity is measured via the inner product of

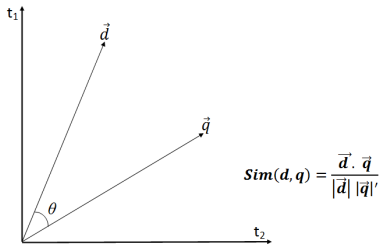


Figure 2.3: The vector space model.

the vectors divided by the product of their *Euclidean lengths*.

$$sim(d,q) = \frac{\overrightarrow{d} \,.\, \overrightarrow{q}}{|\overrightarrow{d}| \times |\overrightarrow{q}|}$$

- ***Probabilistic Model***: A probabilistic model ranks documents in order of their probability of relevance to the user's query. The main idea is known as *Probability Ranking Principle* (PRP), and was first used by Maron and Kuhns [MK60] for literature indexing and searching in a mechanized library system.

  Robertson [Rob97] formulated the principle as a conditional probability. The conditional probability, $P(R|d,q)$, represents the probability of relevance, R, conditioned on the given query q and the document d, where the notion of relevance is regarded as a binary variable. Therefore, all documents are ranked in decreasing order of $P(R|d,q)$ where the goal is to return the most relevant documents. The PRP method provides an optimal ordering of documents if the probabilities of relevance are precisely computed [MRS08]. In practice, the probability of relevance depends on various factors including: (*i*) statistics factors, e.g. term frequencies and document lengths, (*ii*) semantic factors, e.g. the correct meaning of vague terms, and (*iii*) user-dependent factors, e.g. different users may have different judgments of the same query.

  The *Binary Independence Model (BIM)* introduces some assumptions under which estimating the probability $P(R|d,q)$ becomes practical [MRS08]. Here, "binary" means documents and queries are represented as *binary term incidence vectors*. That is, a document vector $\overrightarrow{d}$ is a column of the binary term-document incidence matrix, and a query vector is represented in the same way, i.e. $\overrightarrow{q} = (x_1, ..., x_N)$ where $x_t = 1$ if the term t is present in the query q, and $x_t = 0$ if t is not present. "Independence" means that terms occur in the document independently. Therefore there is no correlation between the terms. Although this assumption is far from correct, it often leads to satisfactory results in practice.

- ***Language Model***: The language model (LM) considers a probability model for each document, called the document's language model. The probability model expresses the distribution of terms in the document. In order to measure the degree of relevance for a document, the language model approach estimates the probability that the query is generated by the corresponding language model. The reasoning behind the language model approach is that the user has a particular

document in mind and formulates a query from this document. The idea of the language model to ranking documents is different from the probabilistic model. Instead of modeling the probability $P(R = 1|d, q)$, the language model first computes a probability distribution M_d for each document d. Then it ranks the document based on the probability $P(q|M_d)$. This approach has provided a novel way of thinking to ad-hoc retrieval, and many extensions have been developed.

After a retrieval model compares the query to each document, it provides a list of relevant documents, called the *result set*. Depending on the retrieval model, an arbitary/ordered list of the documents is displayed to the user.

2.1.3 Result Set

The result set is a list of documents that are determined as relevant to the user's query. Depending on the retrieval model the documents are either in an arbitrary or ordered list. For example, the Boolean retrieval model only separates relevant documents from non-relevant documents. The other retrieval models rank documents with regard to their relevance scores. Clearly documents with higher scores get higher positions in the ranked list. For example, the vector space model ranks documents based on their similarity with the query. A subset of the retrieved results are returned to the user in the form of a display set. The size of the display set is adjusted by constraints such as physical size of the display and user preference. The display set is usually constructed by picking the top k ranked documents.

2.1.4 Relevance Feedback

The result set usually contains a combination of relevant and non-relevant documents. Sometimes, the user cannot find a satisfactory answer to her need, and continues to search by modifying the query. Relevance feedback helps users reformulate the query in order to get better results. The basic procedure is [MRS08]:

1. The user issues a query.
2. The retrieval system[1] returns an initial set of the result set.
3. The user marks some documents as relevant.
4. The system computes a better representation of the information need based on the user feedback.
5. The system displays a revised set of the results.

Sometimes, the process continues for several iterations, and the user will terminate the interaction with either a successful search or failure to find the answer.[2]

2.2 Evaluation

Evaluation aims to either (*i*) measure the absolute performance of an IR system, or (*ii*) the relative performance of several systems. The latter is known as comparative evaluation and commonly used in

[1] A retrieval system refers to a retrieval model. Throughout this thesis retrieval model and retrieval system are interchangeably used.

[2] Relevance feedback is outside of the scope of this thesis and is not discussed further.

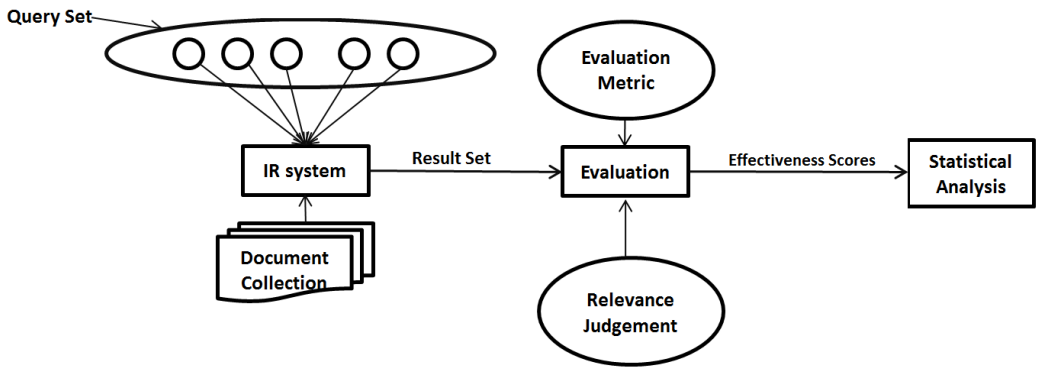


Figure 2.4: Different components of ad-hoc IR evaluation.

information retrieval experiments. A comparative evaluation aims to compare different systems in order to rank their performance or find the best performing system.

The two main aspects of performance are *efficiency* and *effectiveness*. Efficiency is concerned with time and space resources required for indexing, searching and ranking documents. Effectiveness is concerned with how well an IR system meets a user's information need, which is expressed by a query. In this thesis, we focus on effectiveness issues and hereafter the performance refers to the effectiveness aspect of evaluation.[3] In this section, we explain the common experimental design used in Ad-hoc information retrieval task to assess a system's effectiveness.

2.2.1 Experimental Design

In an ad-hoc retrieval task the effectiveness of a system is assessed by an information retrieval test collection used in conjunction with evaluation metrics. A test collection consists of a set of test queries and a collection of documents. Figure 2.4 shows the process of evaluation in an ad-hoc information retrieval task. Each query in a query set is sent to the retrieval system. The retrieval system searches the document collection and returns a set of documents as a result list. Human assessors judge the relevance of the retrieved documents and create relevance judgments. In the simplest case, when relevance is binary, relevance judgments indicate which documents are relevant to the query. The system's effectiveness is computed by using an evaluation metric. The evaluation metric measures how well the result set corresponds to the associated relevance judgments. After evaluating the result sets of all the queries, a statistical analysis (will be shortly discussed in details) is run to investigate the system's overall effectiveness.

2.2.2 Test Collections

Standard experiments in information retrieval are based on the *Cranfield paradigm* of using test collections consisting of documents, topics (search queries), and relevance judgments. The Cranfield paradigm was introduced by Cleverson and his colleagues via the Cranfield 2 experiments [CM97] in the 1960s. It has been improved and extended over the years. The text retrieval conference (TREC), co-sponsored by the National Institute of Standards and Technology (NIST) and U.S. Department of Defense, was started

[3] Performance and effectiveness are used interchangeably throughout the thesis.

in 1992 as a part of the TIPSTER text program. TREC has become the main workshop series designed to foster research in text retrieval. It provides the infrastructure needed for large-scale evaluation of retrieval systems [Voo02]. TREC is currently regarded as the standard resource for IR experiments.

TREC consists of a set of tracks. For each track, a set of particular retrieval tasks are defined. NIST provides an appropriate test collection for a track. Participants run their retrieval systems on the data, and return to NIST a list of top-ranked documents, e.g. 1000 documents per query. The retrieval systems of a participant are usually different configurations of a retrieval model. NIST pools the individual results, judges the pooled documents for correctness, and evaluates the results. The TREC procedure ends with a workshop for participants each year to share their experiences and discuss the relevant issues.

There are also other communities providing test collections. The cross-language evaluation forum (CLEF) provides test collections for cross-language information retrieval (CLIR) where documents are written in a language which is different from the query's language. The Initiative for the Evaluation of XML retrieval (INEX) is another forum that provides test collections for XML information retrieval where the content of documents is a mixture of text, multimedia and meta data.

2.2.3 Effectiveness Metrics

The main task of an effectiveness metric is to measure the relevance of documents retrieved for a query. Two early effectiveness metrics, *recall* and *precision*, were introduced in the Cranfield studies to summarize and compare the search results of different retrieval models. Recall measures how well a retrieval model finds all relevant documents in document collection for a query. It computes the proportion of relevant documents that are retrieved:

$$Recall = \frac{\text{number of retrieved relevant documents}}{\text{the total number of relevant documents}}$$

The definition of recall assumes that for a given query the exact number of relevant documents is known.

Precision measures how well the retrieval model avoids retrieving non-relevant documents. It computes the proportion of retrieved documents that are relevant.

$$Precision = \frac{\text{number of retrieved relevant documents}}{\text{the total number of retrieved documents}}$$

In both metrics relevance is regarded as a binary variable. In addition, both are set-based metrics. That means, the positions of relevant documents in the ranked list do not have any influence on the measurements. Several evaluation metrics have been proposed on the basis of precision and recall for ordered result sets. Precision at rank n, *P@n*, is a rank sensitive metric. It is a form of precision calculated up to the n^{th} rank position. Average Precision (AP) is another rank sensitive metric which is widely used in IR experiments. Calculation of AP involves considering the rank positions at which relevant documents occur, measuring precision at each of the selected rank positions, and finally averaging precision scores. Indeed, AP considers the criteria of precision and recall together for an ordered result set. It is calculated as:

$$AveragePrecision = \frac{1}{|R|} \sum_{i \in R} P@i$$

where R represents the list of rank position of relevant documents, and $|R|$ is the number of relevant documents. The definition of AP regards relevance as a binary variable.

Normalized Discounted Cumulative Gain (nDCG) [JK02] is a graded evaluation metric that is used when the relevance of a document is calibrated into several grades, e.g. $\{0, 1 \text{ and } 2\}$ where 0 means non-relevant, 1 means relevant and 2 means highly relevant. The nDCG metric considers the two following assumptions. First, highly relevant documents are more useful than marginally relevant documents. Therefore, having or missing highly relevant documents has to impact more on the measurement than less relevant ones. Second, the lower the ranked position of a relevant document is, the less important the document is as it is less likely to be visited by users. These two assumptions lead to an evaluation metric that uses graded relevance as a measure of usefulness or gain from examining a document. The gain is accumulated starting at the top of the ranked list and discounted at lower rank positions. If the accumulated gain is divided by the gain that would be obtained by the ideal ordering of documents, it will result the normalized discounted cumulative gain. The nDCG metric is formulated as

$$nDCG = Z_k \sum_{m=1}^{k} \frac{2^{R(m)} - 1}{\log_2(1+m)}$$

where Z_k is the factor normalizing the cumulated gain by the ideal gain that would be obtained at top k retrieved documents. $R(m)$ is the relevance grade of the m^{th} document in the ranked list that is captured from the associated set of relevance judgments.

The Reciprocal Rank (RR) is another rank sensitive metric that is concerned with the rank position of the first relevant document. Thus, it is defined as the reciprocal of the rank position at which the first relevant document is retrieved. Considering a rank threshold (cut off), the provided score is 0 if there is no relevant document at positions below the threshold. The Reciprocal Rank is formulated as

$$RR = \frac{1}{Rank_f}$$

where $Rank_f$ is the rank position of the first relevant document retrieved.

Cooper [Coo68] also introduced the Expected Search Length metric (ESL), which is the average number of documents that must be examined to retrieve a specified number of relevant documents. Another rank sensitive metric that was recently introduced is Rank-biased Precision proposed by Alistair Moffat and Justin Zobel [MZ08]. The main aim of the Rank-biased Precision is to overcome a shortcoming of the Recall and Average precision metric caused by incomplete relevance judgement. Indeed, the problem is an assumption of the Recall metric. The assumption is that the total number of relevant documents for each query is known. However, this assumption is usually not true in practice. In IR experiments, the document collection size usually exceeds millions of documents. In order to find the total number of relevant documents to a query, the relevance of all the documents in a document collection

have to be judged which is economically impossible.

2.2.4 Summarizing Evaluation Results

Once a system's performance is measured across a set of queries, we summarize the results by taking the average of the effectiveness scores. For example, when the effectiveness metric is AP, the arithmetic average of the AP scores measured across the queries is called Mean Average Precision and used as the system's average performance. For n queries MAP is calculated as:

$$MAP = \frac{1}{n}\sum_{i=1}^{n} AP_i$$

MAP is commonly used in IR experiments to compare systems' overall performance. In some experiments, the geometric mean is used instead of the arithmetic mean to emphasize a system's performance one queries for which retrieving relevant documents is not an easy task [Rob06]. For instance, Geometric Mean Average Precision (GMAP) is the geometric mean of the AP scores.

$$GMAP = \exp\frac{1}{n}\sum_{i=1}^{n} ln(AP_i)$$

Another way of summarizing evaluation results is a recall-precision graph. Such a graph provides information about the effectiveness of a retrieval system at a set of recall levels. It summarizes effectiveness over all queries by averaging precision scores at different recall levels. The precision values for all queries at each recall level are averaged and shown as data points in the graph. The recall levels are commonly between 0.0 and 1.0 in increments of 0.1.

As mentioned above, the average is the most common statistic used to summarize evaluation results and compare different retrieval models. In addition, statistical significant tests are used to determine whether a difference between two systems' average performance is statistically significant. Statistical tests that are used in IR experiments are discussed in the next section.

2.2.5 Statistical Significance Tests

The tests of statistical significance have been thoroughly discussed in IR literatures. A group of effectiveness scores measured by an evaluation metric, e.g. AP, across a set of queries is regarded as a statistical sample drawn from a distribution. The distribution represents how effectiveness scores of the population of queries from which a set of queries are chosen to evaluate systems are spread out. The query population includes all possible queries that are issued by users.

In order to run a statistical test to assess a differen between the performance of two systems, a set of queries is first chosen from the query population. Systems search for the chosen queries and return relevant information. Next, the corresponding effectiveness scores are measured by an evaluation metric, and a statistical inference method assesses evidences in favor or against a hypothesis on the distributions from which the effectiveness scores are drawn. The common hypothesis in IR is that the mean value of the two distributions are equal. The methods of inference used to support or reject the hypothesis are known as tests of significance.

Every significance test begins with a *null hypothesis* H_0. In IR, H_0 is the assumption that two systems exhibit the same average performance, e.g. MAP. There is also an *alternative hypothesis*, H_a, which indicates that the difference between the average performance of two systems is statistically significant. The aim of a statistical test is to reject H_0 in favor of H_a. If the test is successful, we can claim that one of the two systems significantly outperforms the other one.

A statistical test provides a probability value called *p-value*. Assuming the two distributions have the same mean, the p-value is the probability that the difference in the average scores occurs by chance. A *significance level*, α, is considered as the threshold for the p-value to examine its significance. If the measured p-value is smaller than the threshold α, the null hypothesis is rejected in favor of the alternative hypothesis. Different significant tests calculate p-value in different ways.

Several significance tests are commonly used in IR evaluation, e.g. Student's paired t-test, Wilcoxon signed rank and the Sign test. The use of these tests is limited by some assumptions made on the data distribution, which is thoroughly discussed in [Hul93]. Furthermore, there are some tests which do not rely on any assumptions of the distribution's form. Bootstrap shift method and Fisher's randomization are two methods proposed in IR [SAC07]. Both approaches test the hypotheses by re-sampling queries from a query set. The bootstrap shift method is defined on a random sample and Fisher's randomization generates samples with permutation. Moreover, Sakai [Sak06] proposed the paired bootstrap hypothesis test which is a combination of the bootstrap shift method and the student t-test.

2.3 Summary

The key components of an ad-hod information retrieval system are: representation, retrieval model, result set and relevance feedback. Each component was discussed in detail. We then discussed the evaluation process in information retrieval experiments. Common test collections, e.g. TREC test collection, are used in IR experiment to assess the performance of different systems. Each test collection consists of a document collection, a query set and a set of relevance judgments. Relevance judgments indicate which documents in a collection are relevant to a query. An evaluation metric, e.g. average precision, is used to measure the performance (effectiveness) of a system against a query. The performance of two systems are compared based on the average of their performance scores measured across a set of queries. Finally, statistical significant tests, e.g. student's paired t-test, are used to assess whether a difference in the average performance of two systems is statistically significant.

Relevance judgments of a test collection are manually created by a set of human assessors. The common test collections in IR experiments contain a large number of documents and queries. As a result, creating a test collection for information retrieval experiments is costly. In the next Chapter, we discuss the practical methods used by IR community to construct cost-efficient test collections.

Chapter 3

Cost Effective IR Test Collections

Information retrieval test collections are used to evaluate the performance of an IR system. Common test collections enable researchers to directly compare their retrieval model with other IR models, or examine the impact of various parameters on the performance of their retrieval models.

A test collection consists of a document corpus, a query set and relevance judgements. Relevance judgments map queries to relevant documents in the corpus. Relevance judgments are used by an IR evaluation metric to measure a system performance against a query. Common IR evaluation metrics, e.g. AP or nDCG, capture two complimentary abilities of an IR system (*i*) ranking relevant documents above the non-relevant ones ("precision"), and (*ii*) identifying all the relevant documents to the given query ("recall"). Hence, a reliable estimate of the performance of a retrieval system might depend on having identified all the relevant documents for a query. This demands to judge all documents in the corpus for a query which is referred to as "complete relevance judgments".

Relevance judgments are manually constructed by a set of human assessors and hence costly. Even for corpora of moderate sizes it is impractical to collect a complete set of relevance judgments for every query in the test collection. In this chapter, we discuss three different methods, namely (*i*) document selection, (*ii*) query selection, and (*iii*) crowdsourcing relevance judgments, that are used to construct an IR test collection and evaluate IR systems under the cost constraints.

3.1 Document Selection

In a typical scenario of IR experiments, a candidate set of queries is compiled, representative of the universe of queries. In case of unlimited resources, we would obtain complete judgments on all documents for every query in the query set. This would give us a gold standard evaluation of the participating systems, and hopefully give us reliable evaluations for future, as yet unseen systems. However, in practice, gathering comprehensive relevance judgments is prohibitively expensive.

To deal with the cost of creating relevance judgments, Sparck-Jones and Van Rijsbergen [SJvR76] suggested to select a particular subset of documents rather than the entire set of documents, in a document corpus, to be judged. For a particular query, they suggested that assessors only judge the documents retrieved at top k rank positions by a set of IR systems that participate in an IR experiment. That is, a set of participating systems add into a pool their top-k, usually k=100, documents retrieved in response to

Table 3.1: number of relevance judgments of TREC test collections [CA05].

TREC	# Queries	# Participating Systems (Runs)	# Relevance Judgments	# Relevant Documents
TREC-3	50	40	97,319	9,805
TREC-4	50	33	87,069	6,503
TREC-5	50	61	133,681	5,524
TREC-6	50	74	72,270	4,611
TREC-7	50	103	80,345	4,674
TREC-8	50	129	86,830	4,728

a query. The pooled document are then delivered to a set of assessors to build the associated relevance judgments. This technique is referred to as *pooling* and widely used by NIST to construct TREC test collections. Once the pooled documents are judged by human assessors, a system's effectiveness score is measured over its top-n ($n > k$), usually 1000, retrieved documents with this hope that many of the documents ranked between $k+1$ and n by this system have been retrieved by other systems at lower rank positions between 1 to k. Also, a retrieved document that was not among the pooled ones was assumed to be non-relevant. The information of several TREC test collections that used the pooling method to gather relevance judgments is shown in Table 3.1.

It has been shown that the number of pooled documents in early TREC experiments, the union of top 100 documents retrieved by each participating system, is sufficient to properly rank the systems performance [Zob98]. However, a considerable amount of relevant documents remain undiscovered. Harman [Har95] built pools of documents ranked between 101-200 for systems in TREC-2 and TREC-3. She reported that a further 11% of relevant documents were discovered in TREC-2 pools and further 21% in TREC-3. Zobel [Zob98] also examined the relationship between the number of identified relevant documents, n, and the cut-off level (depth), p, that is used for the pooling technique. The cut-off level of p determines the set of top p documents in a rank list. He found that the relationship follows a power law distribution.

$$n = Cp^s - 1 \tag{3.1}$$

where C and s are constants. He extrapolated the function for the cut-off level of 500 and concluded that the number of relevant documents would be double that found in a pool of the cut-off level 100. However, he showed that the pool of the cut-off level 100 would be sufficient to measure systems relative performance and rank them correctly.

In early TREC experiments, several alternatives were suggested to build more efficient pools than that was suggested by Sparck-Jones and Van Rijsbergen. Justin Zobel [Zob98] observed that good performing systems that identify more relevant documents in the top-k than other systems receives less benefit from the systems' contribution to the pool, i.e. there is not much overlap between documents retrieved by good systems at ranks above k and the documents retrieved by poor performing systems at ranks between 1 and k. Hence, measuring the effectiveness of good systems at depth n ($n > k$) is likely to underestimate the performance of good systems. Following this observation, he suggested that instead of equally pooling the top-k documents of each system, systems with higher performance should contribute more documents to a pool than lower performing systems.

Regarding the fact that the number of relevant documents varies across queries, Zobel also sug-

gested to set the cut-off level with respect to the number of relevant documents for the given query. If it has become likely that for a query no more documents will be identified, then continuing to judge more documents for that query is a waste of resources.

Cormak et al. [CPC98] proposed a Move-to-Front pooling technique by using a variable number of documents from each system to form a pool. Like Zoble [Zob98], they suggested that good performing systems should contribute more than poor performing systems to the pool. They also proposed an interactive searching and judging (ISJ) method to construct a test collection with fewer judgements compared to the pooling method. ISJ interactively selects a set of documents to build relevance judgments. That is a human assessor submits a predefined query to a retrieval model and judge the relevance of a subset of documents retrieved at the top rank positions. Next, the human assessor reformulates the original query as she learns about the relevant documents and document corpus. The human assessor repeats the process until a predefined number of relevant documents have been identified. Cormak et al. [CPC98] found that with a few hours of work, human assessors could produce as many relevant documents as exist in a document corpus. They suggested ISJ could be used by small research teams to develop effective test collection using minimal resources.

Many of traditional TREC test collections contain only 50 queries. The main reason for not using larger query sets was the cost required to build deep relevance judgments, between 1000 to 3000 documents, for each query. Such amount of judgments per query ensured that recall sensitive metrics, e.g. average precision, are estimated accurately and a test collection provides reliable evaluation results for systems that did not participate in pooling documents.

Voorhees and Buckley [VB02] examined the adequacy of 50 queries to evaluate retrieval systems participating in a TREC experiment. They proposed a measure called "error rate" that quantifies a probability that different query sets of the same size would lead to different rankings of a pair of systems. They empirically modeled the error rate as an exponential function of the absolute difference between two systems' average performances. Evaluating several TREC test collections, they observed that 50 queries would be sufficient to achieve a 5% or less error rate if there was an absolute difference of approximately 0.05 in mean average precision (MAP) scores. A 0.05 absolute difference in MAP corresponded to approximately 15% relative difference with regard to MAP of good performing systems in TREC, which was larger than differences that had generally been observed with TREC experiments. Later Lin and Hauptmann [LH05] investigated weather the empirical error rate function could be derived from statistical principles. They showed that the error rate depends not only on absolute differences but also on the variance of effectiveness scores measured across a query set for a system. They explained that a successful experimental design depends on several factors including a sufficient number of queries, a large enough absolute difference between systems' average effectiveness, and a homogeneous distribution of per-query effectiveness scores, which reduces the variance of the score differences.

Sanderson and Zobel [SZ05] hypothesized that if NIST could evaluate systems by using a larger set of queries, say n', ($n' \gg 50$) and lower cut-off levels, say $k \ll 100$, the assessors' effort to build relevance judgments would be greatly reduced without compromising the accuracy of evaluation. In ad-

Table 3.2: The Kendall-τ rank correlation between the ranking of systems induced by a shallow pool and the ranking induced by a pool depth 100. The data set is TREC-6 and evaluation metric is average precision [CA05].

pool depth	kendall-τ	# judgments	# relevant
1	0.82	1747	460
5	0.899	6652	1216
10	0.93	12209	1747
20	0.964	22937	2477
50	0.981	52874	3575

dition, using many queries with shallow judgments finds more relevant documents than using 50 queries and pools of depth 100. That is because the density of relevance documents at the top of rank lists is higher than the density in lower ranks. Considering TRECs 2-10 test collections, they observed that the number of relevant documents in a pool of depth 10 for a large set of queries is between 1.7 and 3.6 times more than those found when using a smaller query set and a deeper pool. This hypothesis motivated a further work on proposing a new generation of document selection techniques to carefully select a subset of documents for assessments, as well as defining evaluation metrics for partially judged result sets. In the following we introduce these approaches in details.

3.1.1 Incomplete Relevance Judgments and Effectiveness Metrics

Carterette and Alan [CA05] showed that despite measurement errors caused by reducing the pool length, ranking systems based on shallow pools of depth 5, 10 or 20 produce high rank correlations to the ranking of systems induced by a pool of depth 100. The effect of shallow judgments in ranking systems is shown in Table 3.2 for TREC-6. Based on this observation, they proposed a greedy algorithm that incrementally selects a minimal subset of documents that is most informative about the difference between two systems' performance. The algorithm assigns a weight to a document based on a difference that would be provided in effectiveness scores if we assessed this document. Documents are ordered based on their weights and documents with high weights are selected to be judged. Later Carterette et al. [CAS06] proved the validity of this algorithm.

Assessing a subset of documents retrieved by a system causes uncertainty in measuring the corresponding effectiveness scores. Buckley and Voorhees [BV04b] observed that using effectiveness metrics defined on precision and recall, e.g. average precision, to measure systems' performance on incomplete judgments would lead to large measurement errors. They proposed a metric, called *bpref*, and showed that it is more reliable than average precision to measure effectiveness when only a subset of documents are judged. Given a result set, the bpref metric computes a preference relation of whether judged relevant documents are retrieved ahead of judged non-relevant documents. Thus, it is based on the relative ranks of only judged documents. The bpref metric is defined as

$$bpref = \frac{1}{R}\sum_{r} 1 - \frac{\text{number of judged non-relevant documents ranked above } r}{min(R, N)} \tag{3.2}$$

where R is the number of judged relevant documents, N is the number of judged non-relevant documents, and r is a relevant retrieved document. The bpref metric is inversely related to the fraction

of judged non-relevant documents that are retrieved before relevant documents. Although the quantity measured by bpref was different from the quantity of AP, it produces a rank of systems' effectiveness that is highly correlated to the rank induced by AP when measured using the complete set of judgments.

Sakai [Sak07] alternatively suggested to construct the condensed list of documents by discarding all unjudged documents from the original rank list. He applied standard metrics, e.g. AP and nDCG, on the condensed lists and reported that it results in a better solution to incomplete relevance judgments than using bpref. However, later Saki [Sak08] showed that using condensed rank lists leads to a bias in favor of systems that did not participate in the pooling process.

Carterette et al. [CAS06] proposed estimators to accurately approximate standard metrics like $p@k$ and AP when only a subset of documents in a result set are judged. Instead of assuming that unjudged documents are non-relevant as was assumed in traditional TREC experiments, they defined a probability of relevance for each unjudged document. More precisely, they considered each document i to have a distribution of relevance $p(X_i)$. If the document has been given a judgment $j = 0$ or 1 (non-relevant or relevant), then $p(X_i = j) = 1$, otherwise, $p(X_i = 1) = p_i$ and $p(X_i = 0) = 1 - p_i$ where p_i is the probability of relevance computed for an unjudged document. They considered an effectiveness metric as a function of the relevance probability of documents in a result set. For example, the precision at rank position k ($p@k$) was defined as the sum of the probability of relevance of documents ranked between 1 and k. They calculated the expected value and variance of $p@k$ as:

$$E[prec@k] = \frac{1}{k}\sum_{i=1}^{k} p_i \tag{3.3}$$

$$Var[prec@k] = \frac{1}{k^2}\sum_{i=1}^{k} p_i(1-p_i) \tag{3.4}$$

Also, based on the definition of $p@k$ Carterette et al. [CAS06] defined the expected value and variance for the AP metric as:

$$E[AP] \approx \frac{1}{\sum_{i=1}^{k} p_i}(\sum_{i=1}^{k} a_{ii}p_i + \sum_{j>i} a_{ij}p_ip_j) \tag{3.5}$$

$$Var[AP] \approx \frac{1}{(\sum_{i=1}^{k} p_i)^2}\Bigg(\sum_{i=1}^{k} a_{ii}p_i(1-p_i) + \sum_{j>i} a_{ij}p_ip_j(1-p_ip_j) + \sum_{j\neq i} 2a_{ii}a_{ij}p_ip_j(1-p_i) + \sum_{k>j\neq i} 2a_{ii}a_{ik}p_ip_jp_k(1-p_i)\Bigg) \tag{3.6}$$

where i,j and k index over a set of documents retrieved for a query, and if A_i was the rank of document i, $a_{ij} = 1/max\{A_i, A_j\}$. Subsequently, they defined the expected value and variance of average performance, e.g. MAP. Under the assumption that metrics are independent across queries, the mean and variance of MAP, for example, is calculated as:

$$E[MAP] = \frac{1}{n}\sum_{i=1}^{n} E[AP_i] \tag{3.7}$$

$$Var[MAP] = \frac{1}{n^2} \sum_{i=1}^{n} Var[AP_i] \tag{3.8}$$

where n is the number of queries and AP_i is the average precision of query i. According to the central limit theorem and assuming a large number of queries, the random variable of MAP is normally distributed. Hence, the $100 \times (1-\alpha)\%$ confidence interval for MAP is computed as:

$$\left[E[MAP] \pm z_{\frac{\alpha}{2}} \sqrt{Var[MAP]} \right] \tag{3.9}$$

where n is the number of queries and $z_{\frac{\alpha}{2}}$ is a value that satisfies $P(Z \leq z) = 1 - \frac{\alpha}{2}$, where Z is a standard normal distribution. Based on this formulation, it is easy to see that there are two ways to reduce the size of the confidence interval.

In statistical terms, the average precision (AP) of a system can be thought of as a mean of a distribution. The elements of the distribution are a set of relevant documents retrieved by the system and the value of each element is the precision at the relevant document's rank position. Yilmaz and Aslam [YA06] assumed that the distribution is uniform, and randomly sampled a subset of ranked documents to build relevance judgments. They also introduced an estimator, called *InfAP*, to approximate AP. InfAP is approximately an unbiased estimator of AP. However, since top retrieved documents are more likely to be relevant than documents at low ranks, randomly sampling documents from a uniform distribution leads to a high variance in estimating InfAP. Aslam et. al. [APY06] defined a "non-uniform" (biased) sampling distribution to select a subset of documents to build relevance judgments. This sampling method was accurate and resulted in a small variance in estimations, but it was reported to be prohibitively complex to be used in practice. Later Yilmaz et.al [YKA08] suggested partitioning retrieved documents to several strata, and independently sampling a subset of documents from each stratum to build relevance judgements and measure *infAP*. They showed that using stratified sampling is practical and leads to a smaller difference between *infAP* and AP than using uniform sampling. Aslam and Pavlu [AP08] designed a modular approach to evaluate systems on incomplete relevance judgments by separating the sampling from the evaluation module. The sampling module produces a sample of documents in a specific format, e.g. random or stratified sampling, but does not assume a particular evaluation metric is being used. The evaluation module only uses the sample of judged documents to measure a system's effectiveness without any assumption about the sampling distribution. Aslam and Pavlu [AP08] proposed an estimator called *statAP* to approximate AP scores and that was independent of the sampling strategy being used. Given a random sample S of judged documents along with inclusion probability, π_i, the probability that document i is included in sample S, the precision at rank position k is estimated as:

$$prec@k = \frac{1}{k} \sum_{\substack{i \in S \\ rank(i) < k}} \frac{rel(i)}{\pi_i} \tag{3.10}$$

where $rel(i) = 1$ if document i is relevant, otherwise $rel(i) = 0$. Consequently, $statAP$ is defined as:

$$statAP = \frac{\sum_{k \in S:rel(k)=1} \frac{prec@k}{\pi_k}}{\sum_{k \in S:rel(k)=1} \frac{1}{\pi_k}} \tag{3.11}$$

The recent approaches to incomplete relevance judgments, e.g. Carterette et al. [CAS06], Yilmaz et.al [YKA08], and Aslam and Pavlu [AP08], motivated NIST to develop a new series of test collections called Million Query (MQ) track. In 2007, the first MQ test collection was constructed by gathering relevance judgments for about 1800 queries. This was in contrast with traditional TREC test collections, e.g. TREC-8, which usually contains 50 queries only. To deal with the cost of relevance judgments, two document selection algorithms, proposed by Carterette et al. [CAS06] and Aslam and Pavlu [AP08], were used to select a few number of documents per query. On average, 40 documents were selected and judged per query which was considerably smaller than the number of documents that were judged in a traditional TREC test collection. In TREC-8, for instance, on average 1734 documents were selected by the pooling technique and judged per query.

The conclusion of the experiments run on the MQ 2007 test collection was that systems evaluation over many queries with shallow relevance judgments is more cost effective and as reliable as systems evaluation over few queries with deep judgments [CPK+08].

However, acquiring few documents for constructing relevance judgments degrades the *reusability* of a test collection. A test collection is reusable if its relevance judgments would suffice to assess retrieval systems that did not contribute to the document selection process. A new retrieval system may retrieve relevant documents that are not already assessed, and its performance is likely to be misjudged.

In 2009, NIST ran the MQ track for the third time and the main goal was to verify the reusability of large scale test collections where there were thousands queries for which only a few documents were judged (50 per query on average). The main goal was to know whether such a test collection is usable for new systems that did not contribute to the document selection process. The result of comprehensive experiments was that when systems that contribute to the document selection and new systems are the derivatives of the same retrieval model, the test collection is reusable. However, when new systems are derived from retrieval models that are different from those used for participating systems, the test collection is not reusable.

3.2 Query Selection

Query selection is a complementary approach to document selection that is used to reduce the cost of creating relevance judgments. Given a set of queries against which we desire to measure a system's performance, the goal of a query selection approach is to select a subset of them to build relevance judgments and evaluate systems.

Guiver et al. [GMR09] has recently shown that some subsets of queries, known as representative subsets, are particularly good predictors of the systems average performance as measured over the full set of queries. Mizzaro and Robertson [MR07] explored the characteristics of individual queries that were beneficial for systems evaluation. They defined the notion of hubness for queries where a higher hubness

score indicates that a query is better than others at distinguishing the systems retrieval effectiveness. Robertson [Rob11] later showed that the query selection based on the hubness scores alone does not necessarily result in a reliable prediction of the systems rankings. The work by Guiver et al. [GMR09] shows that, indeed, representative query sets that are effective in approximating the systems ranking, comprise of queries that range in their individual ability to predict the systems performance.

Thus, one of the main challenges of the query selection problem is to identify the characteristics of the optimal subset of queries that provide a reliable evaluation of systems and closely approximate the result that would be obtained if we use the full set of queries to evaluate systems. Although there is an abundance of analysis on the document selection problem, little literature is available on how to construct an optimal set of queries for standard test collections. The query selection problem is further discussed in Chapter 4 of this thesis.

3.3 Using Crowdsourcing Experiments to Collect Relevance Judgments

Recently, with the increased capabilities of web services such as Mechanical Turk provided by Amazon[1], it has become feasible to outsource the task of relevance judgments to a large number of people (crowd assessors) rather than assigning the task to a few number of experts who are specifically trained for gathering relevance judgments. This setting provides new opportunities for accomplishing the task through a larger number of assessors which was previously impossible, as well as reducing the time and cost involved in gathering relevance judgments.

Crowdsourcing, in our context, explains how to setup an experiment to gather relevance judgments by using a large number of crowd workers. Using web services like Mechanical Turk, we can formulate the relevance judgments task in terms of Human Intelligence Tasks (HITs). Each HIT contains a set of documents that need to be judged in response to a query. The HITs are presented to the crowd in order to recruit assessors who are willing to engage and provide relevance labels. The cost of the relevance judgments is then captured in the fees paid to the crowd assessors through the micropayment facilities that the crowdsourcing services provide [Alo11].

While still in the early stages, the practices of outsourcing the relevance judgment tasks are evolving and practitioners are investigating the benefits and the drawbacks of the crowdsourcing approach [NR10]. Issues such as the assessors' agreement among the highly skilled editorial staff are now expanded to include a number of factors that are directly related to the unique crowdsourcing paradigm, including the mechanism for qualifying workers, providing incentives, controlling behavior and label quality, and designing and promoting tasks. While crowdsourcing holds the promise of achieving the scale of relevance judgments in a considerably shorter period of time, the cost of engagement and quality assurance are key elements that need to be carefully planned and managed. We further discuss these issues of crowdsourcing experiments in Chapter 8.

[1] www.mturk.com

3.4 Summary and Directions

The increased size of document corpora and query sets has made the cost of relevance assessments one of the main challenges in creating IR test collections. To deal with the cost of gathering relevance judgments three approaches were introduced, namely (*i*) the document selection approach that minimizes the number of documents that need to be judged per query, (*ii*) the query selection approach that minimizes the number of queries used to evaluate system and (*iii*) the crowdsourcing experiments that outsource the relevance judgment task to a large number of crowd assessors rather than assigning the task to a few expert assessors.

Although there is a large body of literature on document selection approaches, little work is available on query selection and crowdsourcing relevance judgements is still on its early stages. We describe the query selection problem in details in Chapter 4. Also, the issues related to the crowdsourcing experiments are discussed in Chapter 8.

Chapter 4

The Query Selection Problem

Effective evaluation of information retrieval systems requires building test collections that contain a set of queries and associated relevance judgments. Relevance judgments are manually constructed and can be costly. In real world settings, the budget is constrained and imposes a limit on the number of relevance judgments that can be acquired. Hence, algorithms that can be used to reduce the number of judgments are needed.

We focus on query selection as a mechanism for reducing the cost of building test collections. We develop a theoretical framework for query selection. We assume that relevance judgments are available for all the queries under consideration and show how the query selection can be formulated as an optimization problem. The mathematical formulation provides valuable insights into the characteristics that the optimal subset of queries holds. Since the optimization problem is computationally intractable, we introduce two algorithms that provide approximate solutions. We demonstrate the effectiveness of the two query selection algorithms by using two TREC test collections, namely TREC-8 Ad-hoc and TREC 2004 Robust tracks.

4.1 Introduction

Modern test collections are large, comprising billions of documents and thousands of queries that require relevance judgments in order to calculate retrieval effectiveness metrics. One of the main problems of such a test collection is the cost of building associated relevance judgments. Much recent work has been devoted to constructing cost-efficient test collections with the primary focus on reducing the number of documents to be judged per query, e.g. [CAS06, YA06, AP08]. This approach is known as *document selection* which was widely discussed in Chapter 3. We, on the other hand, focus on minimizing the number of queries that are required to reliably evaluate the performance of a set of systems, which is known as *query selection*.

Finding the minimum number of queries that are required to reliably evaluate the performance of a set of systems has been one of the main challenges since early TREC experiments, e.g. [VB02, SZ05]. Previous work, e.g. [CS07, WMZ08b], considered this problem purely as a statistical sampling question. Under the assumption that queries are randomly selected the goal was to run power analysis [Bil95] to investigate the minimal sample size of a random subset to obtain a statistically robust evaluation result

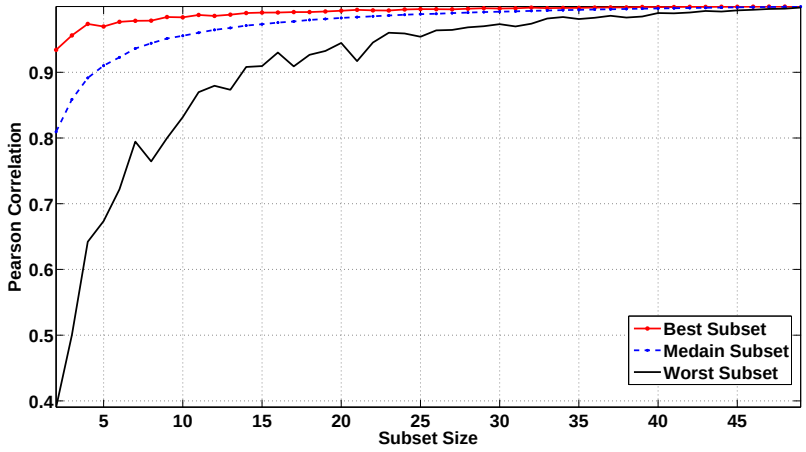


Figure 4.1: Pearson linear correlation of the best, median and worst subsets of various sizes, chosen from 1000 random subsets, on TREC-8 test collection.

with a certain confidence. Though there is an abundance of analysis on various aspects of random sampling, little literature is available on alternative approaches that require smaller number of queries without compromising the accuracy of evaluation results.

Recently, Guiver et al. [GMR09] showed that query subsets of a particular size vary in predicting systems overall performance that is measured using the full set of queries. They also showed that there is a particular subset of queries that enables a precise prediction of systems' overall performance. The size of the subset was about $\frac{1}{3}$ the size of a random sample of queries to achieve the same accuracy in evaluation. We repeated one of their experiments using TREC-8 test collection that comprises 50 queries. For each subset size between 2 and 49 we randomly selected 1000 subsets of the 50 queries. For each chosen subset we computed the mean average precision (MAP) of each system in TREC-8. To measure prediction accuracy of a subset we computed Pearson linear correlation between the set of MAP scores computed using the chosen subset and the MAP scores that were computed using the full set of queries.

Figure 4.1 represents the resulted Pearson correlation versus the subset size for three types of subsets: (*i*) the *best* subsets that exhibit the maximum Pearson correlation, (*ii*) the *median* subsets that achieve the median Pearson correlation among the 1000 subsets, and (*iii*) the *worst* subsets that exhibit the minimum Pearson correlation. The best subset of size 6 achieved over 0.95 Pearson correlation. However, the median and worst subsets needed at least 20 and 31 queries to obtain the same Pearson correlation. Figure 4.2 represents almost similar results for Kendall-τ correlation that was used to compute the closeness between the two systems' ranking induced by a subset of queries and the full set of queries.

This experiment and those conducted by Guiver et al. [GMR09] validate the hypothesis that some queries or query subsets are better than others at predicting systems' overall performance, and that with the right choice of queries, accurate predictions is obtainable by using a subset of queries. Therefore, it is possible to reproduce the results of exhaustive evaluation of systems over many queries with a smaller

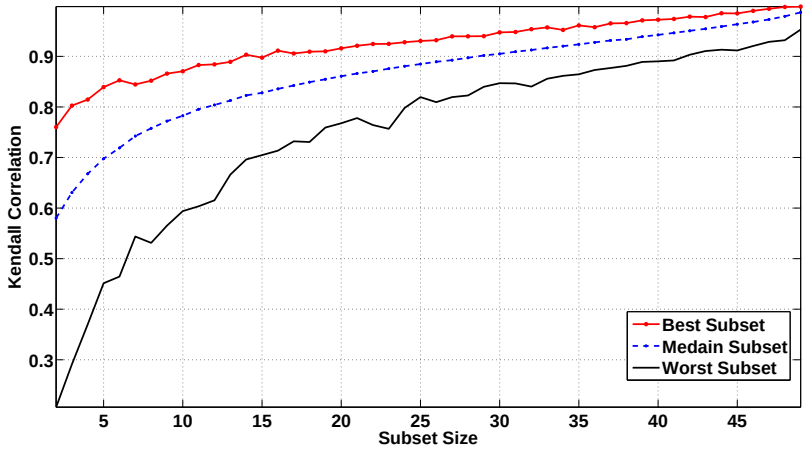


Figure 4.2: Kendall-τ correlation of the best, median and worst subsets of various sizes, chosen from 1000 random subsets, on TREC-8 test collection.

set of queries. However, the characteristics of chosen queries that make them representative of the full set are still unclear. Also, how the query selection is implemented in practice was not addressed in previous work.

We first define the query selection as an optimization problem. We assume that relevance judgments are available for all the queries and develop a theoretical framework for the query selection problem. The mathematical formulation implies that the best subset of queries should satisfy two properties. These are that (*i*) the selected queries are least correlated with one another, and (*ii*) the selected queries should have strong correlation with the remaining queries. We briefly remark that correlation between two queries refers to their similarity in evaluating systems, not in the statistical or semantic correlation between their terms.

Since selecting the optimal subset of queries is a computationally intractable problem, we approximate the solution by using two various algorithms. We evaluate the two algorithms by comparing the systems' ranking for the subset of queries with the ranking over the full set of queries. We report the results in terms of Kendall-τ and Pearson correlation coefficients and by using two TREC test collections, namely (*i*) TREC-8 consisting of 50 queries, and (*ii*) TREC 2004 Robust track consisting of 249 queries.

4.2 A Framework for Query Selection

Let Q be the population of queries, and S be the space of all search systems.[1] We assume that for each of the queries in Q we have an associated effectiveness score, measured by a metric e.g. AP, for each system in S. By averaging the effectiveness scores across queries, we compute the average (expected) performance of a system.

In practice, both the number of queries and the number of systems are finite. Consider $Q_n \subset Q$ with n known queries together with a set of l known systems, $S_l \subset S$. We can consider Q_n and S_l to represent the queries and participating systems in our test collection. We remark that the l participating

[1]That is, not only systems that participated in pooling documents to build relevance judgments, but also systems that did not participate to the pooling process.

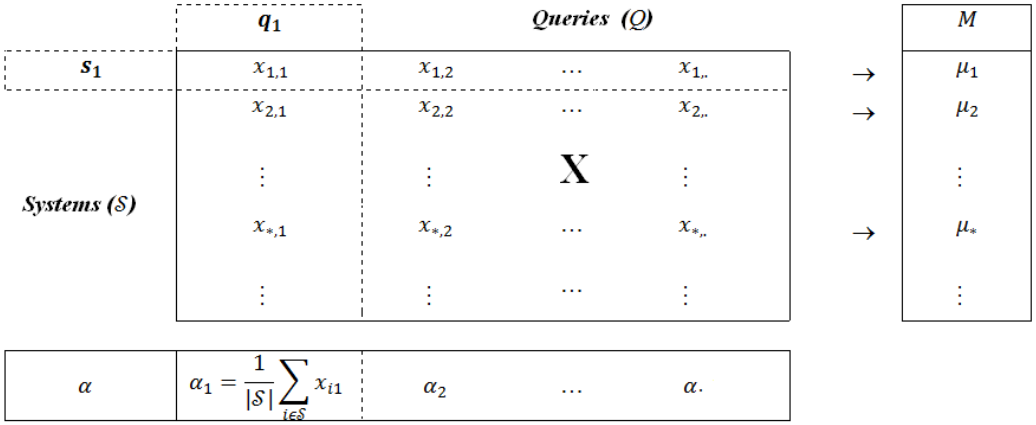


Figure 4.3: The matrix X representing the performance metric of a set of systems S against a set of queries Q.

systems contribute to the pooling process and a system that is not in S_l is referred to as a new system, or a previously unseen system. The combination of the l systems and n queries forms a $l \times n$ matrix $X \in R^{l \times n}$. Each row represents a system, and each column a query. An entry, x_{ij}, in X denotes the performance of system i on query j. We refer to any column of the matrix X as a *query-systems* vector, the values of which represent the performance of each system for a specific query. We also define a column vector $M \in R^{l \times 1}$, which represents the average of all query-systems vectors, as seen in Figure 4.3. The values of M indicate the average performance of individual systems over all queries. Thus, if the individual elements, $x_{ij} (1 \leq i \leq l$ and $1 \leq j \leq n)$, measure average precision (AP), the elements of M, $\mu_i (1 \leq i \leq l)$, represent mean average precision (MAP).

Now, let $\Phi = \{j_1, ..., j_m\}$ be a subset of $\{1, 2, ..., n\}$ with $1 \leq m \leq n$ and Q_Φ be the corresponding query subset. We define $M_\Phi \in R^{l \times 1}$ as the column vector comprising the average performance of systems for the subset of queries, Q_Φ. The aim of a query selection method is to find a subset of queries of a particular size, m, such that the corresponding column vector M_Φ closely approximates the vector M.

The approximation can be quantified using the mean squared errors between the elements of M and M_Φ, if the similarities in the absolute values of performance scores are of interest. Alternatively, we can use Kendall-τ correlation if the similarity in the ranking of systems is of interest, or Pearson linear correlation if the similarities in the relative performance scores are of interest.[2]

In common experiments in IR, we are usually interested in relative comparisons of the performance of IR systems and use Pearson linear or Kendall-τ correlation as our evaluation measure. We also focus on the correlation measures when modeling the query selection problem.

4.3 A Formal Model for the Query Selection Problem

We develop an optimization model for the query selection problem using Pearson Linear correlation as it is amenable to mathematical optimization. However, we use both Kendall-τ and Pearson correlation

[2]The mathematical definition of the three measurement is give in Appendix D.

as our final evaluation measures for comparing the evaluation results produced by the full and a subset of queries.

The Pearson correlation between the column vectors M and M_Φ is

$$\rho_\Phi = \frac{cov(M, M_\Phi)}{\{var(M)var(M_\Phi)\}^{\frac{1}{2}}} \tag{4.1}$$

To compute the Pearson correlation, ρ_Φ, we need to compute the variances and covariance of M and M_Φ. Let Σ denote the $n \times n$ covariance matrix of performance matrix X. Also, let σ_{ij} denote the $(i,j)^{th}$ element of Σ and be the covariance between the i^{th} and j^{th} query-systems vectors. Consequently, σ_{ii} denote the variance of the i^{th} query-systems vector. Remember that a query-systems vector is a column of matrix X, and is therefore the vector of effectiveness scores across systems for a single query. The variance of M can be computed in terms of the covariance matrix Σ as

$$var(M) = n^{-2} e^T \Sigma e$$

where $e \in \{1\}^{n \times 1}$ is a column vector of n ones. Similarly, the variance of M_Φ is computed as

$$var(M_\Phi) = m^{-2} d^T \Sigma d$$

where $d \in \{0 \text{ or } 1\}^{n \times 1}$ is a binary vector that indicate the set of m selected queries. Therefore, if query j is selected $d_j = 1$, otherwise $d_j = 0$, also $\sum_{j=1}^{n} d_j = m$.

To compute the covariance between M and M_Φ we consider an unknown system that is randomly sampled and denote $x \in R^{1 \times n}$ as the associated row performance vector in X. The system's average performance computed based on x and the full set of queries is

$$\mu = n^{-1} x e$$

Also the systems' average performance based on the subset of m queries, Q_Φ, and x is

$$\mu_\Phi = m^{-1} x d \tag{4.2}$$

The covariance between $\hat{M}_\Phi$ and M is then

$$cov(M_\Phi, M) \equiv cov(\mu_\Phi, \mu) = m^{-1} n^{-1} cov(xd, xe) = \\ m^{-1} n^{-1} d^T cov(x^T, x) e = m^{-1} n^{-1} d^T \Sigma e$$

where $xd = d^T x^T$ and

$$cov(x^T, x) = E\big\{ \big(x - \alpha\big)^T \big(x - \alpha\big) \big\} \equiv \Sigma$$

where $\alpha \in R^{1 \times n}$ is the mean row vector of matrix X, see Figure 4.3. The j^{th} element of α is the mean

of the j^{th} query-systems vector.

Substituting for the variances and covariance of M and M_Φ, the Pearson linear correlation is

$$\rho_\Phi = \frac{(d^T \Sigma e)}{\left\{(e^T \Sigma e)(d^T \Sigma d)\right\}^{\frac{1}{2}}} \tag{4.3}$$

Formally, we seek a subset of queries, Q_Φ, that maximizes ρ_Φ. Reordering the equation 4.3 for correlation above we have

$$\gamma_\Phi \equiv (e^T \Sigma e)^{\frac{1}{2}} \rho_\Phi = \frac{e^T \Sigma d}{(d^T \Sigma d)^{\frac{1}{2}}} \tag{4.4}$$

Selecting queries for the subset Φ maximizing ρ_Φ is equivalent to selecting queries that maximizes γ_Φ since $(e^T \Sigma e)^{\frac{1}{2}}$ is a constant. Suppose, for example, the query subset Q_Φ only contains a single query j, so that $\Phi = \{j\}$. Then

$$\gamma_\Phi = \frac{\sum_{i=1}^{n} \sigma_{ij}}{(\sigma_{jj})^{\frac{1}{2}}} \tag{4.5}$$

where $\sum_{i=1}^{n} \sigma_{ij}$ is the j^{th} column total of Σ, and σ_{ij} is the $(i,j)^{th}$ element of Σ, i.e., the covariance between i^{th} and j^{th} query-systems vectors, and $\sigma_{jj}^{\frac{1}{2}}$ is the standard deviation of the j^{th} query-systems vector. In general, the optimal subset Q_Φ of a particular size is the one with the maximum value of

$$\max_{\Phi} \gamma_\Phi = \frac{\sum_{j \in \Phi} \sum_{i=1}^{n} \sigma_{ij}}{(\sum_{i,j \in \Phi} \sigma_{ij})^{\frac{1}{2}}} \tag{4.6}$$

Equation 4.6 provides valuable insight into the query selection problem. In order to maximize γ_Φ we would like the denominator to be small and the numerator to be large.

Consider the denominator $(d^T \Sigma d)^{\frac{1}{2}}$. Remember that the covariance matrix, Σ, is fixed and represents the covariance between the n query-systems vectors. An element, d_j, of the binary vector, d, is one if query j belongs to the subset. To minimize the denominator, we must choose the m queries that are least correlated with one another. This is equivalent to maximizing the information we derive from each query in the subset. Conversely, if the query-systems vectors are perfectly correlated, then all the queries provide the same information and we may as well have a subset of size one.

Now consider the numerator, $e^T \Sigma d$. This is maximized if the subset of query-systems vectors has high correlation with the rest of the queries. This is also intuitively clear. After all, if the subset of query-systems vectors is completely uncorrelated with the remaining query-systems vectors, then this subset can provide no prediction of how systems will perform on the remaining queries. Assuming that an evaluation on the full set of query-systems vectors is a gold standard, the objective encodes a preference for subsets that have a strong correlation with the full evaluation. Note that this correlation is between query-systems vectors. It is not a statistical correlation between terms in queries, nor is it a semantic correlation between queries. This is an important distinction. The queries “cat” and “dog” are neither statistically nor semantically correlated. However, the query-systems vector for “cat” may be strongly correlated with the query-systems vector for “dog”, and thus the query-systems vector for “cat” is able to predict the corresponding performance for “dog”.

4.4 Query Selection Algorithms

Finding the subset of queries that maximizes Equation 4.6 belongs to the family of subset selection problems that are NP-hard [WEST03]. Brute-force options are available for small n but impractical when n is large. We are seeking a method that takes as input a $l \times n$ effectiveness matrix X representing the evaluation of l reference systems on n queries, and produces as output a subset of m queries. In the subsections 4.4.2 and 4.4.3 we describe two algorithms that efficiently find approximate solutions. However, before that we first explain the random query sampling method that is being widely used in IR community, and hence is considered as the baseline in our experiments.

4.4.1 Random Query Sampling

The common way of selecting a subset of queries for IR test collections is random sampling [ACA$^+$07]. In this method, there is no criterion for selecting a subset and all queries are given the same chance to be selected.

4.4.2 Greedy Query Selection

A forward greedy algorithm can be used to approximately find the optimal subset. That is, when m=1, the optimal subset is the query whose query-systems vector obtains the maximum value of the equation 4.5. For every $m > 1$ we use the best subset of size m-1 and select the m^{th} query from the queries indexed in Φ^c (the complement set of Φ) that maximizes the equation 4.6.

This greedy algorithm is fast and tractable but is not guaranteed to find the best subset since the best subset of size m does not necessarily contain all the queries selected for the best subset of size m-1 [GMR09].

The greedy algorithm can accept any measures as its objective. Therefore, when the ranking of systems is of interest, we can directly use Kendall-τ with the greedy algorithm. At each iteration, we select a query that its combination with the previously selected queries results the maximum Kendall-τ between M and M_Φ.

4.4.3 Convex Optimization

An improvement of the query selection model proposed in Section 4.3 is to seek arbitrary linear combinations of effectiveness scores of a query subset, rather than just taking unweighted averages. The average performance μ_i of system i in S_l can be expressed as a linear combination of the effectiveness scores x_{ij}, associating a coefficient with each query j. Let $\beta \in R^{n\times 1}$ be a vector of n real values. Then the linear combination is expressed as

$$\mu_{i\beta} = \sum_{j=1}^{n} \beta_j x_{ij} = x_i \beta \tag{4.7}$$

where x_i is the i^{th} row of matrix X.

We define $M_{\Phi\beta} \in R^{l\times 1}$ as the vector of the l systems' average performance computed using a linear combination of a query subset, Q_Φ. Thus, the goal of the query selection is to set the β so that the correlation $\rho_{\Phi\beta}$ between M and the corresponding $M_{\Phi\beta}$ is maximized. The correlation $\rho_{\Phi\beta}$, expressed

in terms of the covariance matrix Σ, is

$$\rho_{\Phi\beta} = \frac{(\beta^T \Sigma e)}{\left\{(e^T \Sigma e)(\beta^T \Sigma \beta)\right\}^{\frac{1}{2}}} \tag{4.8}$$

Following the formulation in Section 4.3 the optimization function is expressed as

$$\max_{\beta} \gamma_{\Phi\beta} = \frac{e^T \Sigma \beta}{(\beta^T \Sigma \beta)^{\frac{1}{2}}} \qquad \text{subject to} \quad \| \beta \|_0 \leq m \tag{4.9}$$

where $\| \ . \ \|_0$ is the L_0 norm constraint that simply counts the number of non-zero elements in β and controls the size of the subset. Therefore, if a query j is selected, $\beta_j > 0$; otherwise $\beta_j = 0$. [3] Solving the optimization in Equation 4.9 is a subset selection problem as we need to test the $\beta > 0$ coefficients for any query subsets of size m. Thus, finding the optimal solution is NP-hard [WEST03]. We slightly change the optimization function in Equation 4.9 to form it as a convex optimization for which computationally efficient solutions are available [BV04a].

The maximum value of Equation 4.9 is *approximated* by the minimization function that is expressed in a quadratic form [MoWMMRCC67]

$$\min_{\beta} \ \frac{1}{2} \beta^T \Sigma \beta - e^T \Sigma \beta \qquad \text{subject to} \quad \| \beta \|_0 \leq m \tag{4.10}$$

To minimize Equation 4.10 we use convex relaxation that replaces the above minimization function with a convex function that admits tractable algorithms. Note that the optimization function in Equation 4.10 is not convex because of the L_0 norm constraint. We alter this constraint to convert Equation 4.10 to a convex form. To do so, we replace L_0 norm constraint by an L_1 norm constraint that is the closest convex form to L_0. Choosing the optimal subset is now based on solving the following optimization function

$$\min_{\beta} \ \frac{1}{2} \beta^T \Sigma \beta - e^T \Sigma \beta \qquad \text{subject to} \quad \| \beta \|_1 \leq C \tag{4.11}$$

where $\| \ . \ \|_1$ is the L_1 norm that returns the sum of absolute values of the elements in β. Also, C is a positive real value between 0 and $+\infty$. The optimization in Equation 4.11 is convex and can be efficiently solved by the quadratic programming algorithms [Mur88, SFR07] to generate the optimal subsets of size $\{1, 2, ..., n\}$ as C varies from 0 to $+\infty$.

4.5 Estimations of Covariance Matrix

In practice, the values of the mean vector α and covariance matrix Σ are unknown to us because the space of all systems S, including both participating and unseen systems, is unknown. Hence, the mean row vector, α, and covariance matrix, Σ, are estimated using the sample of size l participating systems.

[3] The role of the β vector in Equation 4.9 is similar to the role of the binary vector d in Equation 4.4.

4.5.1 Random Sampling of Systems

Considering the sample $\{x_1, ..., x_l\}$ of multivariate scores the estimators of α and Σ are given by

$$\hat{\alpha} \equiv \bar{x} = l^{-1} \sum_{i=1}^{l} x_i \tag{4.12}$$

$$\hat{\Sigma} = (l-1)^{-1} \sum_{i=1}^{l} (x_i - \bar{x})^T (x_i - \bar{x}) \tag{4.13}$$

The estimators above are unbiased if the set of l participating systems is uniformly sampled from the population of systems S. In this case, if l is large and the sample of participating systems forms a diverse set of retrieval systems, we obtain reliable estimations of α and Σ. The unbiased estimators ensure that all the l participating systems contribute equally to select a subset of queries or estimate weight scores β. Therefore, when a new system is randomly sampled from S, the unbiased estimators find a subset that provides reliable evaluation results.

4.5.2 Non-random Sampling of Systems

In practice, new systems may not be randomly selected from S. They may, in fact, be variations and extensions of the pervious systems with high performances. In this case, allowing all the participating systems to contribute equally to selecting queries may not result in the best choice. Instead, better performance may be achieved by selecting queries based on participating systems that are similar to the new system.

We denote $\{p_1, p_2, ..., p_l\}$ as a set of weights assigned to the l participating systems such that $\sum_{i=1}^{l} p_i = 1$. The weight p_i indicates the degree of contribution for the i^{th} participating system in selecting queries. The unbiased estimators of α and Σ are then

$$\hat{\alpha} = \sum_{i=1}^{l} x_i p_i$$

$$\hat{\Sigma} = \frac{1}{1 - \sum_{i=1}^{l} p_i^2} \sum_{i=1}^{l} (x_i - \hat{\alpha})^T (x_i - \hat{\alpha}) p_i$$

For instance, if we assume that new systems will have high performance, we can weight the participating systems such that higher performing participating systems contribute more to the selection of queries. In this case, we use unbiased estimators of a weighted sample of systems to approximate α and Σ.

In our experiment, we describe a simple selection method for weights p_i and investigate the use of the corresponding estimators in a real situation of IR experiments in which new systems are expected to obtain high performance.

4.6 Experiments

We evaluated the performance of the three query selection algorithms introduced in Section 4.4. We selected subsets of varying size m, using each of the three query selection algorithms. The quality of a selected subset was then assessed in terms of (*i*) *accuracy* and (*ii*) *generalization*. Accuracy is concerned with how well a subset of queries can reproduce the relative performance of the participating systems when measured against the full set of queries. Generalization is concerned with how well the selected subset of queries can reliably evaluate a set of new systems, again compared to the whole set of queries. Before proceedings, we first describe the experimental data.

4.6.1 Experimental Setup

Normally, organizations participating in TREC register as sites and submit a number of experimental runs for evaluation. These runs often represent variations on the system's settings. For our purposes we should consider runs as IR systems, taking a special care when considering runs from the same site. In our experiments, we used (i) the TREC-8 test collection and (ii) the Robust TREC 2004 track. The TREC-8 test collection consists of 50 queries (topics), 39 sites with 129 runs of which 13 runs are manual and 116 runs are automatic. Automatic runs automatically create queries but manual runs use queries that are created by human experts. We use the TREC-8 test collection throughout the experiments to create a heterogenous data set for the purpose of our generalization experiments as explained in Section 4.6.3. The Robust TREC 2004 track consists of 249 queries, and 14 sites with 110 automatic runs. The query set of the Robust TREC 2004 track contains 49 new queries (a 50th was removed because no relevant documents were found), also the 50 queries from the TREC 2003 Robust task, and 150 queries from the TREC-6, TREC-7 and TREC-8 test collection. We use the TREC 2004 Robust test collection in our experiments because it's query set is a combination of five different query sets that makes it one of the largest query sets among the TREC test collections, and hance suitable for the query selection task. In TREC-8 and Robust tracks, between 1000 to 3000 documents were judged per query and metrics, e.g. AP and $P@10$, were used to measure systems' performance. In our experiments, we used AP to measure systems' performance.

To assess the accuracy and generalization of a query selection algorithm, we partitioned the set of all systems in a TREC test collection, i.e. experimental runs, into 'participating' and 'new' (unseen) systems. In order to ensure that new systems were truly different from the participating ones, we held out as new systems not only individual runs but the entire set of runs from the same site. Furthermore, during computation of performance metrics, we removed the documents that were uniquely retrieved by the held-out (new) systems. The reduced pool was used to measure the performance of the participating systems and to construct the associated performance matrix X.

We used a repeated random sub-sampling technique to split systems into participating and new systems. At each trial, we randomly selected 40% of sites, and labeled their runs as new systems. Choosing 40% of sites ensures us that a sufficiently large number of runs are set aside to test the generalization of a query selection method. The remaining runs were treated as participating systems and used to build the pool of judged documents for system evaluation. The performance of participating systems were mea-

sured and used to construct the performance matrix X and associated covariance matrix Σ. Because the new systems were selected by random sampling, we used the unbiased estimators, introduced in Section 4.5.1, to compute Σ. We then applied the three query selection algorithms, namely, *random*, *greedy* and *convex*, to select a subset of queries. For the random query sampling method we reported the average of 1000 random trials.

We repeated the process of random partitioning over 50 trials and reported their average results. The 50 trials of sampling ensured that the runs of each site were at least assigned once to the participating set and once to the new set. The K-fold cross validation [Koh95a] was another option for partitioning. However, the advantage of the random sub-sampling over the K-fold cross validation was that the proportion of the participating/new split was independent of the number of iterations (folds). Consequently, a considerable subset of runs, 40% of the total runs in a TREC dataset, was set aside as new systems to obtain robust evaluation results on generalization experiments.

4.6.2 Accuracy

For the full set of queries we constructed the associated document pools using the set of participating systems. We also provided the performance matrix X using AP metric, and the corresponding covariance matrix Σ. Next, we selected a subset of queries of size m using one of the query selection algorithms, and computed the corresponding vector M_Φ. In order to measure the accuracy of the selected query subset, we computed the Pearson linear correlation and Kendall-τ rank correlation between M_Φ and the corresponding vector M. The elements of vector M were the participating systems' true MAP computed using the full set of relevance judgments in the original test collection.

The accuracy of the three query selection algorithms on the TREC 2004 Robust track with 249 queries is shown in Figure 4.4 for Pearson linear correlation. The results are represented for the subset sizes between 1 and 50. We reported the averages of 50 trials as the average results for the greedy and the convex methods. The averages of 1000 random trials were also reported as the average results of the random method. We also considered the 95% confidence interval of the averages to detect significant differences between the query selection methods. For instance, for the subset of 10 queries the average of 50 Pearson correlation scores obtained by the greedy method was 0.94 and the associated standard deviation was 0.03. Thus the 95% confidence interval was $\left[0.94 \pm 1.96 \times \frac{0.03}{\sqrt{50}}\right]$. The confidence intervals in Figure 4.4 are shown by error bars around the averages. Thus, significant differences in performance are detected if the error bars of two methods do not coincide. For all the subsets sizes, the accuracy obtained by both the greedy and the convex methods significantly outperformed the accuracy of the random sampling. Also, the accuracy of the convex method was superior to the accuracy of the greedy method across subsets of various size.

We also repeated the experiment for the Kendall-τ rank correlation. The greedy algorithm computed the Kendall-τ between M and M_Φ to find the best subsets. The results are shown in Figure 4.5 for various subsets between 1 and 50. Again both the greedy and the convex methods significantly outperformed the random method. However, as opposed to the results observed in Figure 4.4 for Pearson correlation, the greedy algorithm consistently outperformed the convex method across various subsets.

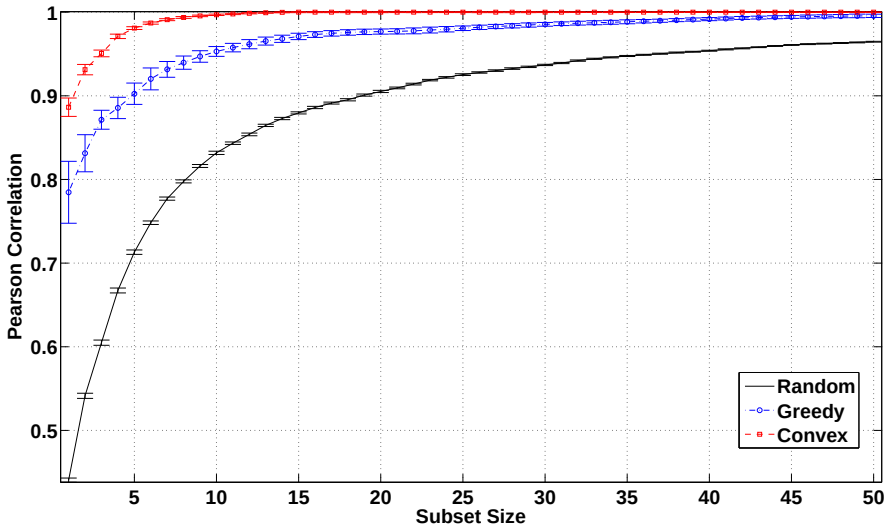


Figure 4.4: Accuracy of the three query selection methods: random, greedy and convex, measured by Pearson correlation on TREC 2004 Robust track with 249 queries. The greedy method used the optimization function in Equation 4.6 to find the best subset of queries.

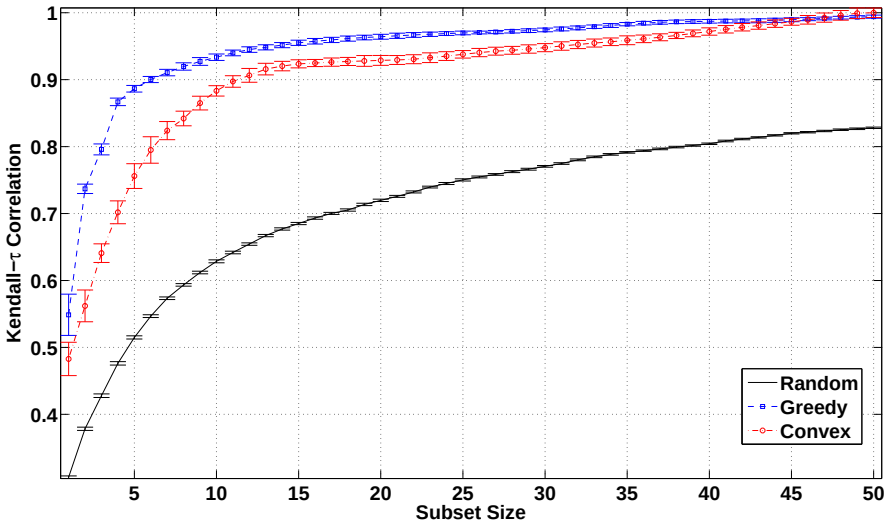
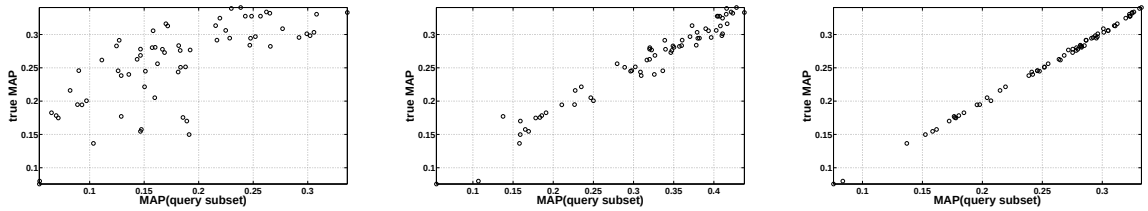
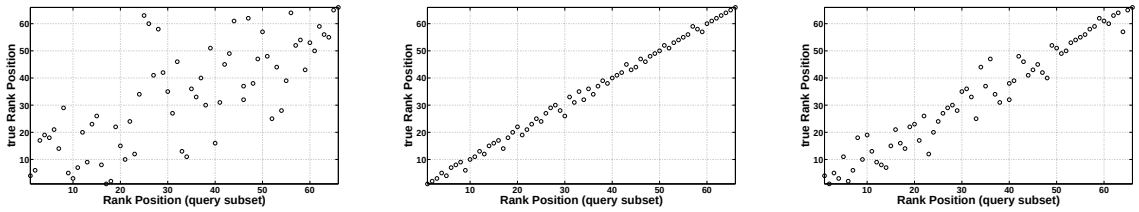


Figure 4.5: Accuracy of the three query selection methods: random, greedy and convex, measured by Kendall-τ rank correlation on TREC 2004 Robust track with 249 queries. The greedy method directly used the Kendall-τ between M and M_Φ to find the best subset of queries.



(a) Random (b) Greedy (c) Convex

Figure 4.6: Scatter plots of the systems' MAP calculated for a query subset of size 5 and systems' MAP of the full set of queries. The systems are the participating systems in one of the random trials of our experiment in Section 4.6.2



(a) Random (b) Greedy (c) Convex

Figure 4.7: Scatter plots of the systems' ranking calculated for a query subset of size 5 and systems' ranking of the full set of queries. The systems are the participating systems in one of the random trials of our experiment in Section 4.6.2

According to the formulation in Section 4.4.3 the convex method is optimized for Pearson correlation. Thus, the convex method may not be able to obtain the optimal results for Kendall-τ correlation. In contrast, the greedy algorithm is directly optimized for Kendall-τ and is expected to obtain near to optimal results for the participating systems.

Also, we randomly picked participating systems in one of the trials and drew, as an example, the scatter plots of the systems' MAP scores measured using the full set of queries versus the systems' MAP scores measured by a subset of 5 queries that is chosen by one of the three query selection methods. The results are shown in Figure 4.6.

Similarly, Figure 4.7 represents the corresponding scatter plots of the systems' ranking using the full set of queries versus the systems' ranking measured by a subset of 5 queries that is chosen by one of the three query selection methods.

4.6.3 Generalization

The new systems' performance were computed using the document pools that were constructed by the participating systems. The corresponding vector M_Φ was computed based on a query subset that was chosen by one the three algorithms. To measure generalization of a query subset, we computed the Pearson Linear correlation and Kendall-τ rank correlation between the M_Φ and the corresponding vector M. The elements of M represented the new systems' true MAP computed the full set of relevance judgments in the original TREC test collection. In the convex method the vector M_Φ weighted each query equally, i.e. the value of β coefficients, calculated as a part of the solution to Equation 4.11 were

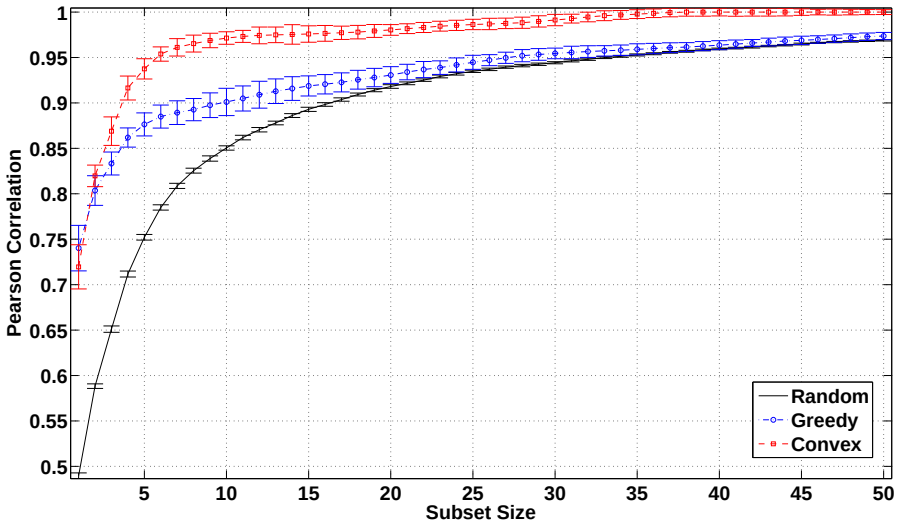


Figure 4.8: Generalization of the three query selection methods measured by Pearson correlation on TREC 2004 Robust track with 249 queries.

ignored. We notice that we only used the β coefficients to select a subset of queries and discarded them when computing the M_Φ vector for the new systems. We do this because the sample mean of AP scores of selected queries is an unbiased estimator of MAP of a new system calculated over the full set of queries, and amongst all the unbiased estimators, it has the smallest variance [Hub74].

The generalization of the three query selection methods on TREC 2004 Robust dataset is shown in Figure 4.8 for Pearson correlation and Figure 4.9 for Kendall-τ correlation. The subset size varies between 1 and 50. The averages of 50 trials were reported as the average results of the greedy and the convex methods. As seen in Figure 4.8, the Pearson correlation obtained by the convex method significantly outperformed the Pearson correlation of the greedy method for all the subset sizes between 3 and 50. While the convex method consistently outperformed the random results, the greedy method failed to perform better than the random method for subset sizes bigger than 20.

Almost similar results were obtained in Figure 4.9 for Kendall-τ correlation. We note the greedy algorithm directly computed the Kendall-τ between M_Φ and M to find the best subsets. As opposed to the results in Figure 4.5, the convex method significantly outperformed the greedy method for all the subsets between 5 and 50. For instance, the convex method obtained 0.9 Kendall-τ correlation after selecting 35 queries. However, the greedy method required at least 76 queries to obtain 0.9 Kendall-τ correlation.

Considering the results in Figure 4.8 and 4.9 as the size of the subset increased, the greedy method over-fitted to precisely evaluating the participating systems and consequently lacked generalization. That is to say that, as opposed to the convex method, the greedy method is unable to recover from choices it made earlier on since it is committed to using a query in all sizes once it has been chosen at an initial iteration. Comparing Figures 4.9 and 4.5, all the three query selection algorithms have lower Kendall-τ correlations for generalization, indicating that evaluations of new systems are likely to be less accurate.

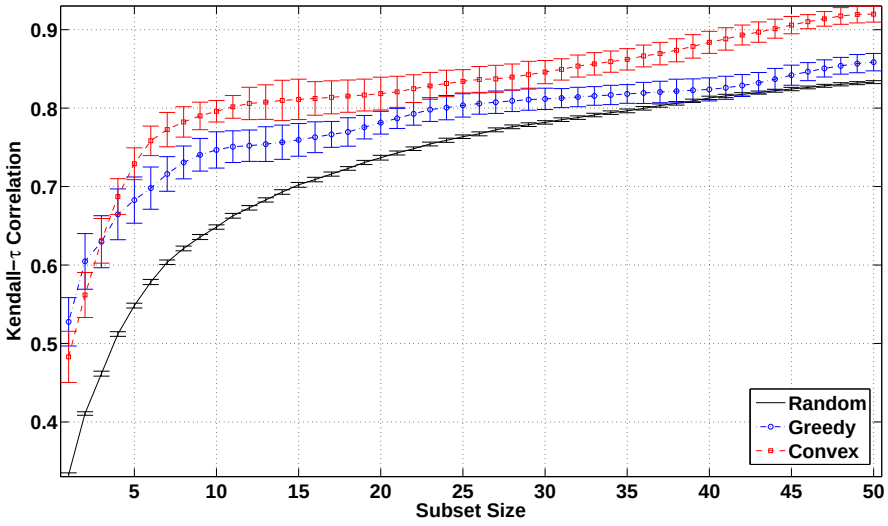
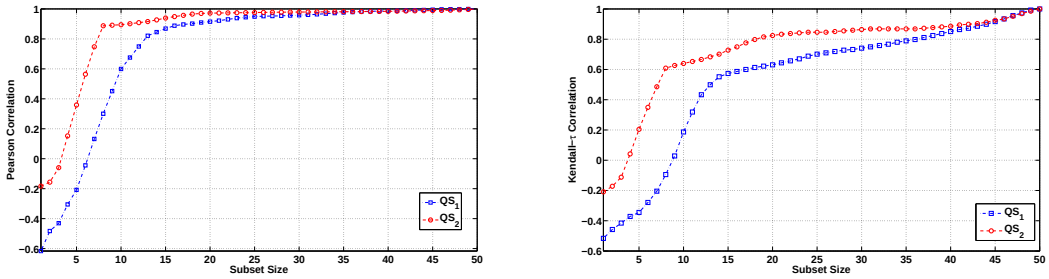


Figure 4.9: Generalization of the three query selection methods measured by Kendall-τ correlation on TREC 2004 Robust track with 249 queries.



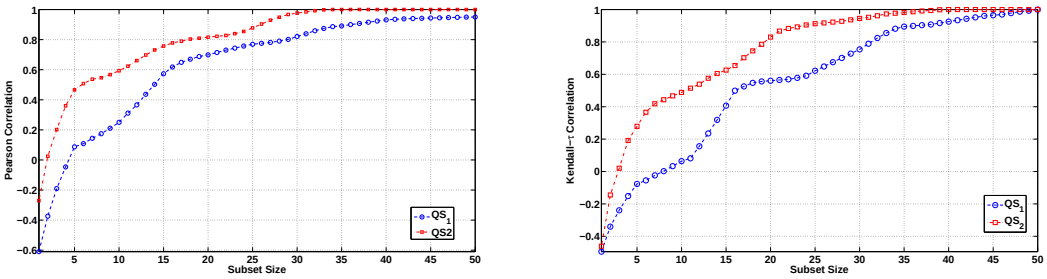
(a) (b)

Figure 4.10: The performance of the greedy algorithm on evaluating the 13 manual runs in TREC-8 when: (*i*) the covariance matrix Σ is approximated based a uniform sample of automatic runs (QS_1), and (*ii*) Σ is approximated based on a weighted sample of the automatic runs (QS_2).

In the next chapter, we show that how this issue can be at least partially alleviated by acquiring a few additional judgments based on the documents that are solely retrieved by the new systems.

4.6.4 Evaluating a Non-Random Sample of New Systems

So far we assumed that the new systems were randomly selected from the space of all systems. We now consider a more *realistic* scenario of IR experiments in which new systems are indeed not a random sample and completely different from participating systems, known as heterogeneous systems in previous work, e.g. Robertson [Rob11]. To form a heterogenous data set we follow the previous work, e.g. [Rob11], and use the TREC-8 test collection. The TREC-8 test collection consists of 129 runs (systems) of which 116 runs are automatic and 13 runs are manual. Both the automatic and manual runs use the same set of retrieval models. The only difference is that for the manual runs queries are formulated by human experts while for the automatic runs queries are formulated by machine, and without human intervention. In practice, manual runs usually outperform automatics ones. In TREC-8 the 11 best



(a) (b)

Figure 4.11: The performance of the convex query selection algorithm on evaluating the 13 manual runs in TREC-8 when: (*i*) the covariance matrix Σ is approximated based a uniform sample of automatic runs (QS_1), and (*ii*) Σ is approximated based on a weighted sample of the automatic runs (QS_2).

performing runs are all manual and their performance measured by MAP are statistically significantly better than the remaining runs. For the purpose of our experiment, we considered the 13 manual runs as new systems and the rest as participating systems. We considered two variants of the query selection model namely QS_1 and QS_2. The QS_1 method used the unbiased estimators, explained in Section 4.5.1, to approximate the mean vector α and the covariance matrix Σ. The QS_2 method used the unbiased estimators for a weighted sample of systems, as explained in Section 4.5.2. Thus, when using QS_2, the participating systems contributed non-uniformly in selecting queries. The intuition was that, since new systems were likely to perform better than participating systems, we could achieve better generalization of new systems, if we preferentially weighed highly performing participating systems.

We used a simple weighting function to weigh the participating systems for the QS_2 method. After pooling documents by using the full set of participating systems we selected a subset of k participating systems with the highest MAP scores. If the i^{th} system was among the selected ones, the corresponding weight was $p_i = \frac{1}{k}$, otherwise $p_i = 0$.

In our experiment, we set $k = 30$ because (i) the MAPs of the top 30 best performing participating systems were significantly larger than the MAPs of the remaining, and (ii) the set of 30 systems comprised a sufficiently large sample of well performing systems.

The performance of QS_1 and QS_2 are shown in Figures 4.10a and 4.10b for Pearson correlation and Kendall-τ correlation. We used the greedy query selection algorithm for both QS_1 and Q_S2 to iteratively select a subset of queries. As seen, the performance of QS_2 is superior to the performance of QS_1 for both Pearson and Kendall-τ correlation across various subsets.

Similar results were obtained when the convex method was used as the query selection algorithm for both QS_1 and QS_2. The results are shown in Figure 4.11a and 4.11b for Pearson and Kendall-τ correlation. As seen, the weighted sampling of systems caused a better approximation of the covariance matrix Σ for selecting a subset of queries that reliably evaluate the set of manual runs.

4.7 Summary and Directions

We defined the query selection problem and showed how it is formulated as an optimization problem. The mathematical formulation showed that an optimal subset satisfies two properties: (*i*) the selected queries should be least correlated with one another, thereby maximizing the information we gain from each, (*ii*) the selected queries should have strong correlation with the remaining queries, as without this correlation there is no predictive capability.

Finding the globally optimum subset of queries is NP-hard and hence computationally expensive. We introduced two algorithms, namely greedy and convex, that could be used in practice to approximate the optimal solution. We compared the performance of the greedy and convex methods against the random query selection based on accuracy and generalization. Accuracy was concerned with how well a subset of queries can reproduce the relative performance of the participating systems. Generalization was concerned with how well the selected subset of queries could provide reliable evaluation results for a set of new systems. Our experimental results on two TREC test collections, namely TREC-8 and TREC 2004 Robust test collections, showed that the convex method is superior to the random selection and the greedy method when Pearson correlation was used as the evaluation metric. When the evaluation metric was Kendall-τ, the greedy algorithm outperformed the convex method in ranking participating systems. However, the greedy algorithm laked generalization and was unable to reliably evaluate a set of new systems.

So far, we have introduced a theoretical framework for query selection and discussed the properties of the optimal subset. We also conducted some retrospective experiments to evaluate various query selection algorithms. In the next two chapters, we will explore the applications of query selection and show how it is used in practice to reduce the cost of IR test collections.

Chapter 5

The Reusability of a Test Collection

The state-of-the-art process for constructing test collections involves using a large number of queries and selecting a set of documents, submitted by a group of participating systems, to be judged per query. However, the initial set of judgments may be insufficient to reliably evaluate the performance of new as yet unseen systems. We show how a query selection algorithm can be used as a budget-constrained optimization method to expand the set of relevance judgments as new systems are being evaluated.

We assume that there is a limited budget to build additional relevance judgements. From the documents retrieved by the new systems we create a pool of unjudged documents. Rather than uniformly distributing the budget across all queries, we first select a subset of queries that are effective in evaluating systems and then uniformly allocate the budget only across these queries. Experimental results on TREC 2004 Robust dataset demonstrated the superiority of this budget allocation strategy.

5.1 Introduction

Gathering relevance assessments has an associated cost which, in its simplest form, depends on the number of queries and the number of documents per query that need to be assessed. However, the cost is not the only consideration when creating effective test collections. The accuracy and reusability of the test collections are also very important. A test collection is accurate if the participating systems' performance are precisely evaluated. In addition, a test collection is reusable if has no inherent bias that might affect evaluation of new as yet unseen systems.

Following the belief that a larger query set is desirable, the Million Query track of TREC 2007 [ACA$^+$07] was the first to include thousands of queries. The Million Query track used two document selection algorithms, proposed by Carterette et al. [CAS06] and Aslam et al. [APY06], to acquire relevance judgments for about 1,800 queries. The experiments on this test collection showed that a large number of queries with a few judgements (i) resulted in an accurate evaluation of participating systems, and (ii) was more cost-effective than evaluation conducted by fewer queries with more judgements. However, due to the small number of documents assessed per query, the reusability of such a test collection still remains questionable. Indeed, Carterette et al. [CKPF10] demonstrated that the Million Query track of TREC 2009 is not usable for assessing the performance of systems that did not participate in pooling documents.

In particular, a test collection may not be reusable if a new system, in response to queries in the test set, retrieves many documents that are not in the document pool. In this situation, (*i*) the previously unjudged documents must either be judged non-relevant [Voo02], (*ii*) the new documents are assigned a probability of relevance and new systems' performance are measured by using metrics designed for incomplete relevance judgments, e.g. MTC [CAS06], or (*iii*) additional user relevance judgments must be obtained for these documents. Assuming the documents are non-relevant potentially biases the test collection. Only future systems that behave like the original participating systems will be evaluated accurately [CKPF10]. Assigning a probability of relevance may cause a high uncertainty in evaluation when there are a large number of unjudged documents for new systems [CGJM10], and acquiring additional user judgments can be expensive.

We assume a limited budget is available to build additional relevance judgements for previously unjudged documents retrieved by new systems. We examine whether it is better to uniformly allocate the budget across all queries, or select a subset of queries and allocate the budget only to the selected queries to get deeper judgments per query at the same cost. We report our experimental results on TREC 2004 Robust test collection and show the advantages of using the query selection approach in enhancing the reusability of a test collection.

5.2 Expanding Relevance Judgements

We begin with the assumption that a system can be reliably evaluated and compared with other systems if we manually assess a significant portion of the document corpus or, at least, a large number of documents retrieved by each individual system. This assumption is valid when the recall sensitive metrics ,e.g. AP and Recall, are used to measure a system's performance.

Therefore, if new systems return many new (unjudged) documents, the current relevance judgements are insufficient to reliably assess their performance. In this situation, we assume that there is a limited budget to build relevance judgements for a subset of the new documents. How should we spend the limited budget to acquire additional relevance judgments? We could consider all queries and use a document selection algorithm to pool a few documents per query. Alternatively, we could select a representative subset of queries that closely approximates systems' overall performance, and allocate the budget only to the selected queries. The final solution is likely to include elements of both these approaches. The document selection problem has been widely discussed in previous work, e.g. [CAS06, AP08]. We assume the pooling method [SJvR76] is used to select documents at the query level and restrict our attention to the effects of choosing queries.

Selection of the subset is strongly related to the query selection problem defined in Chapter 4. Thus, ideally, we would identify a minimal subset of queries that still enables a reliable evaluation of the existing and new systems. Furthermore, the gain from reducing the number of queries can be redirected to increase the number of documents judged per query. Our hypothesis is that, given a fixed budget, a smaller but representative set of queries with a greater number of judged documents per query will increase the accuracy of ranking new systems.

In the next section, we show how the convex query selection is formulated as a budget-constrained

optimization to select a subset of queries. The convex query selection method accepts the budget as a constraint and adaptively selects a subset of queries. Thus, the number of selected queries depends on the budget that is available for expanding relevance judgments.

5.3 Budget-Constrained Query Selection

We denote Ω as the cost of building relevance judgements for all previously unjudged documents that are returned by new systems. Also B denotes the limited budget ($0 \leq B \leq \Omega$) that is available to build additional relevance judgements. We also define $\beta \in [0,1]^{N\times 1}$ which contains real values bounded between 0 and 1 such that if j^{th} query is selected, $\beta_j > 0$, otherwise $\beta_j = 0$. The budget constraint is defined as a linear combination of β coefficients to control the number of selected queries:

$$\sum_{j=1}^{N} \beta_j \leq \frac{B}{\Omega} \tag{5.1}$$

We now consider the convex query selection formulated in Section 4.4.3. The L_1 constraint is replaced with the linear budget constraint (5.1) to form our budget-constrained optimization

$$\min_{\beta} \ \frac{1}{2}\beta^T \Sigma \beta - e^T \Sigma \beta \qquad \text{subject to} \quad \sum_{j=1}^{n} \beta_j \leq \frac{B}{\Omega} \tag{5.2}$$

where Σ is the covariance matrix of a performance matrix X that contains the performance scores of l systems against n queries. Also, $e \in \{1\}^{n\times 1}$ is a vector of n ones.

To solve the budget-constrained convex optimization the quadratic programming algorithm [Mur88, SFR07] can be used to generate the optimal subsets of size $\{1, 2, ..., n\}$ as the budget B varies from 0 to Ω. Hence, we select all queries for which β_j is non-zero, and by varying the budget B we control the number of queries in the subset.

5.4 Experimental Evaluation

Our experimental investigations were performed using the TREC 2004 Robust dataset consisting of 249 topics (queries), and 14 sites with a total of 110 runs. We considered runs as search systems, taking special care when considering runs from the same site.

For the purpose of our experiments, the set of all experimental runs were partitioned into *participating* and *new* systems. In addition, in order to ensure that new systems were truly different from the participating ones, we held out as new systems not only individual runs but also the entire set of runs from the same site. Furthermore, during computation of performance metrics, we removed the documents that were uniquely retrieved by the new (held-out) systems from the pool.

5.4.1 Experimental Setup

We assumed a fixed budget is available to collect new relevance judgments. We examined two methods for allocating the budget across queries. In the first method, the resources were equally spread across all queries. For example, if the budget could cover only 200 new judgments and there were 100 queries,

we judged two new documents per query. In the second method, we selected a subset of queries and then allocated the budget equally across them. We used the budget-constrained query selection method, introduced in Section 5.3, to select a representative subset of queries.

We first randomly selected a subset of sites and used their experimental runs as participating systems. We then analyzed the held-out sites and distinguished between those sites that performed similarly to the held-in sites, i.e. there was considerable overlap in the documents retrieved by these sites and the held-in sites, and those sites that were very different from the held-in sites. To do this we applied the reusability measure proposed in [CGJM10] to measure the extent to which the corresponding pooled documents covered the documents retrieved by the held-out systems. For each held-out system and each query we considered the ranked list of documents and computed the average reuse (AR),

$$AR(q) = \frac{1}{judged(q)} \sum_i \frac{judged@i(q)}{i}$$

where $judged@i(q)$ was the number of judged documents in the top-i results of the held-out system for query q, and $judged(q)$ was the total number of documents judged for query q. In addition, we defined the mean average reuse (MAR) as the average of AR values for a system over the full set of queries.

We separated held-out sites into two groups based on the average of the MAR scores of their runs. Those with high MAR across runs that could be evaluated using the existing relevance judgments, and the second group with runs that had low MAR and thus required additional relevance judgments in order to be evaluated. The first group of runs formed the *auxiliary set* of systems, and the others were considered as *new systems*. The auxiliary set was added to the set of participating systems to form the performance matrix X and aid the selection of queries. We also used the new systems to evaluate the different resource allocation methods. The full experiment is as below:

1. Pick s_1 sites at random. The runs of these sites are treated as participating systems.

2. For each query, construct the *initial* pool of top-k_0 documents retrieved by participating systems and build associated relevance judgments. Compute the performance matrix X for the participating systems and the full set of queries.

3. Compute the MAR for the runs that did not participate in the pooling. Average the MAR scores across the runs from the same site and produce average reuse score for each site.

4. Pick s_2 sites with the smallest scores and treat their runs as *new systems*. The remaining runs are *auxiliary systems* that can be evaluated with the existing relevance judgments. Their performance values are added to the performance matrix X. Note, however, that the auxiliary systems do not contribute to the document pool.

5. Given a budget B, select a subset of m queries using the budget-constrained convex optimization method.

6. Acquire additional relevance judgments in one of two ways:

(a) **Subset**: For each of the m selected queries assess an additional k_1 documents contributed by the new systems where k_1 is adjusted based on B.

(b) **Uniform**: For each of the n queries[1] assess an additional k_2 documents contributed by the new systems where $m \times k_1 = n \times k_2$.

7. Add the newly judged documents to the initial pool and compute the effectiveness scores for the new systems.

5.4.2 Experimental Results

We applied the above steps across 10 trials. In each trial, we randomly chose a different set of participating sites, s_1=1, 3 or 5. The runs of the remaining sites were partitioned into auxiliary and new systems based on their reusability scores. We considered the s_2 lowest scoring sites and chose their runs to be new systems, where $s_2 = 3$, 6 or 8. The auxiliary sets comprised 5, 6 or 7 sites. To construct the initial pools we considered the top-k_0 documents from each participating system, where $k_0 = 10$ or $k_0 = 30$. Assuming a fixed budget, $B = \{1, 3 \text{ or } 5\} \times 10^4$ and a performance matrix X composed of participating and auxiliary systems, we used the convex optimization method to select a subset of queries. As B increased, the number of selected queries also increased. In our experiments, the size of the subsets varied between 14 to 237 with a median of 69.

Table 5.1 compares the performance statistics for the Robust 2004 track test collection before and after acquiring new relevance judgments in 12 different experimental configurations. The values given in the table are Kendall-τ scores – averaged over 10 trials – between the ranking of new systems induced by the "initial pool" (containing top-k_0 documents returned by participating systems) or one of the two resource allocation methods ("uniform" and "subset") and the ranking induced by MAP scores that are measured over the full set of queries and by using the original pools (TREC *qrels*). Also, p^+ counts additional pairs of systems that are correctly ordered by the subset method when compared to the number of pairs correctly ordered by the uniform method. In addition, Ω is the number of judgements needed to build relevance judgements for all previously unjudged documents that are returned by new systems in a pool of depth 100.

We note that if the difference in average performance scores of two systems is not statistically significant, it is completely reasonable that they may be ordered differently when evaluated over a subset of queries. Having such tied systems in a test set increases the probability of a swap and consequently decreases Kendall-τ. This is because the Kendall-τ is not able to distinguish between pairs of systems with and without significant differences. This is the case in Robust track test collection in which about 30% of pairs are ties, when measured by a paired t-test at significance level 0.05. In Table 5.1, the Kendall-τ scores in parentheses are calculated by only considering the pairs of systems with a statistically significant difference in MAP.

The positive effect of increasing the number of sites s_1 that contribute to the document pool, can be observed from the experiments 1, 7 and 10 for which s_1 is varying from 1 to 5, with $B = 1 \times 10^4$.

[1] The size of the full set of queries is denoted by n.

Table 5.1: Results for TREC 2004 Robust runs evaluated by MAP. The first six columns report experimental parameters. The next three columns report the Kendall-τ of ranking new systems in the basis of the initial pool and each of the two budget allocation methods. The last column (p^+) counts additional pairs of systems that are correctly ordered by the "subset" method against the "uniform" method. The values in parentheses are measured by only considering pairs of new systems with a statistically significant difference.

exp.#	s_1	s_2	k_0	Ω	B	$\frac{B}{\Omega}$	Kendall-τ						p^+	
							initial pool		uniform		subset			
1					10,000	0.06			0.54	(0.63)	0.6	(0.66)	95	(50)
2	1	8	10	163,842	30,000	0.18	0.42	(0.49)	0.57	(0.66)	0.64	(0.70)	110	(63)
3					50,000	0.31			0.61	(0.69)	0.68	(0.74)	111	(79)
4					10,000	0.09			0.66	(0.74)	0.73	(0.79)	53	(44)
5	1	6	30	107,817	30,000	0.28	0.54	(0.61)	0.70	(0.77)	0.77	(0.81)	62	(42)
6					50,000	0.46			0.75	(0.80)	0.82	(0.85)	62	(46)
7					10,000	0.1			0.70	(0.76)	0.76	(0.83)	53	(51)
8	3	6	10	104,580	30,000	0.29	0.60	(0.65)	0.75	(0.81)	0.82	(0.87)	62	(53)
9					50,000	0.48			0.82	(0.80)	0.89	(0.89)	71	(79)
10					10,000	0.19			0.87	(0.91)	0.90	(0.95)	7	(11)
11	5	3	10	53,535	30,000	0.56	0.70	(0.77)	0.92	(0.94)	0.96	(0.98)	11	(8)
12					50,000	0.93			0.99	(1.0)	0.98	(1.0)	-2	(0)

In addition, increasing s_1 or k_0 increases the average reuse scores of held-out sites and, consequently, reduces the number of new systems and the amount of Ω. This can be seen from the experiments 1-9. Diversifying the set of participating systems by increasing s_1 while keeping k_0 constant causes a bigger improvement in Kendall-τ than the opposite, i.e., increasing k_0 and keeping s_1 constant. This is demonstrated by the experiments 4 and 7 where $s_2 = 6$ and $B = 1 \times 10^4$. This result is consistent with observations by Carterette et al. [CGJM10] that a higher diversity of participating systems results in a better ranking of new systems.

In experiments 1-9 where the total cost, Ω, is considerably bigger than the available budget, B, the subset method significantly outperforms the uniform method. In practice, we usually prefer to obtain Kendall-τ=0.9 by spending a minimum budget. Our subset allocation method obtains τ=9.0 in experiment 10 where an additional B=10,000 budget is spent to gather relevance judgments. However, the uniform method reaches τ=0.9 in experiment 11 where B=30,000 budget is required which is 20,000 documents more than the subset method. As B approaches Ω, the amount of improvement decreases such that in the last experiment ($s_1 = 5$ and $\frac{B}{\Omega} = 0.93$) the Kendall-τ obtained by the uniform method is bigger than the Kendall-τ for subset. We note that, as B approaches Ω, the number of selected queries gets closer to the total number of queries in the test collection. Therefore, when $\Omega \approxeq B$, the difference between the performance of subset and uniform method is negligible.

5.5 Summary

We considered the problem of expanding the relevance judgements of a test collections in order to better evaluate the performance of new systems. Given a fixed budget, we investigated whether it is better to uniformly allocate the budget across all the queries in the test collection, or only to a subset of queries. Our hypothesis was that a smaller but representative set of queries with a greater number of judged documents per query increases the accuracy of ranking new systems.

The hypothesis was tested using the TREC 2004 Robust track. For a fixed budget, we compared how well new systems were ranked, based on a uniform allocation across (i) all queries and (ii) a subset of representative queries. The subset of queries was selected by using the convex query selection method. The budget constrained was added to the convex query selection method to control the number of selected queries.

A variety of different experimental configurations were tested, which (i) varied the number of participating sites (1, 3 or 5), (ii) the number of new sites (3, 6 or 8), (iii) the size of the top-k_0 documents contributing to the initial pool, and (iv) the budget available ($B = \{1, 3 \text{ or } 5\} \times 10^4$ additional relevance judgments). When B was much smaller than the required budget, Ω, to build complete relevance judgements, allocating the budget uniformly across a subset of queries performed better than uniform allocation across all queries. As B approached Ω the difference between two methods became negligible.

Chapter 6

Uncertainty-Aware Query Selection

We extend the query selection framework introduced in Chapter 4 by relaxing the assumption that relevance judgments are available before selecting queries. We show how the extended optimization framework can be used in practice to reduce the cost of IR test collections. Since the query selection optimization is computationally intractable, we devise an iterative query selection algorithm that provides an approximate solution. Our method selects queries iteratively and assumes that no relevance judgments are available for the query under consideration. Once a query is selected, the associated relevance assessments are acquired and then used to aid the selection of subsequent queries.

We demonstrate the effectiveness of the algorithm on two TREC test collections as well as a test collection of an online search engine with 1000 queries. Our experimental results show that the queries chosen by our method produce a ranking of systems' performance that is better correlated with the actual ranking when compared to queries selected by the existing baselines. We also investigate how the selected query subset generalizes to (*i*) new unseen systems and (*ii*) changes to the evaluation metric. We show that our iterative algorithm can be modified to improve generalizability in both cases.

6.1 Introduction

The query selection problem was defined in Chapter 4 to reproduce the results of an exhaustive evaluation of systems by using a representative subset of queries. A query selection framework was modeled based on the assumption that relevance judgments are available for all queries under consideration and systems' performance scores are known. However, reducing the cost of a test collection is possible only if we can select queries before collecting relevance judgments.

We assume that a large set of queries have been initially compiled against which we desire to measure the performance of a set of systems. However, the available budget only permits collecting relevance judgments for a subset of queries. Our goal is to find a subset of queries that most closely approximates the results that would be obtained if one provided relevance judgments for the full set of queries.

We extend our query selection model by relaxing the assumption that relevance judgments are available prior to selecting a query. In contrast to previous work which is mostly retrospective and assumes some relevant judgments are available for each query, e.g. Guiver et al. [GMR09], Mizzaro and

Robertson [MR07], Hauff et al. [HHdJA09] and Robertson [Rob11], our model is designed to work in practice and does not require the existence of relevance judgments for a query that is not selected yet.

We explicitly model the uncertainty in the retrieval effectiveness metrics that are introduced by the absence of relevance judgments. The mathematical formulation shows that an optimal subset should satisfy a number of properties. These are that (*i*) selected queries have a low correlation with one another, thereby maximizing the information we gain from each, (*ii*) selected queries have strong correlation with the remaining queries, as without this correlation there is no predictive capability, and (*iii*) the total uncertainty associated with the selected queries is small.

Since selecting the optimal subset of queries is a computationally intractable problem, we approximate the solution by an iterative query selection process. The algorithm starts by selecting the first query with no information about relevance judgments. However, once this query is selected, associated relevance judgments are acquired and used to assist with the selection of subsequent queries.

Specifically, at each iteration we use previously selected queries and associated relevance judgments to train a classification method that estimates the relevance of documents pooled for each of the unselected queries. Using the classifier's outputs we compute the relevance probability of pooled documents which in turn are used to estimate the values of a performance metric, e.g. average precision, and corresponding approximation variance which we refer to as *uncertainty*.

We evaluate our method by comparing the systems ranking for the subset of queries with the ranking over the full set of queries. We report the results in terms of Kendall-τ and Pearson correlation coefficients and show that the query sets chosen by our models are significantly more effective than those selected by considered baselines for ranking systems.

Query subset selection methods may exhibit poor performance when estimating the performance of previously new (unseen) systems [Rob11]. We conduct experiments to investigate how our method generalizes to new systems. We show that the iterative algorithm can be modified to improve generalizability. Additionally, we consider the query selection problem for the use of multiple metrics. In our experiment we show that a subset selected based on a particular metric may not provide a reliable evaluation result when another metric is used to measure systems' performance. Thus we modify our query selection algorithm to select a query subset that enables reliable evaluation across multiple metrics.

In summary, our contributions in this chapter are threefold. Namely, (*i*) we provide a theoretical model for query selection that explicitly models uncertainty in retrieval effectiveness scores, (*ii*) we develop an iterative algorithm that efficiently implements our theoretical model in practice, and (*iii*) we modify the iterative algorithm to investigate how the selected query subset generalizes to (1) new unseen systems and (2) changes to the evaluation metric. We show that the modified algorithm improves generalizability in both cases.

6.2 Query Selection Principles and Notations

We consider a set of l system and n queries. When relevance judgments are available, the performance of the l systems against the n queries are represented by a performance matrix $X \in R^{l \times n}$, as shown in Figure 6.1, where x_{sq} shows the performance score, e.g. the average precision, of the s^{th} system on

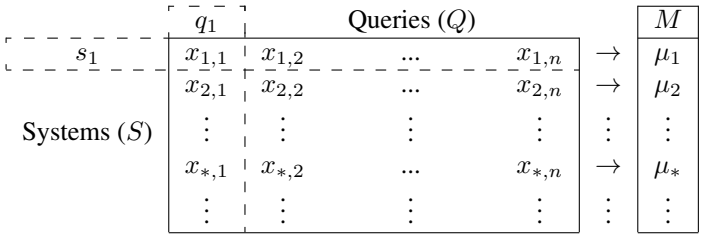


Figure 6.1: The true performance matrix X for a set of system systems and a set of queries. Each entry indicates the system performance score based on the available relevance judgments.

the q^{th} query. We also consider a column vector $M \in R^{l\times 1}$, as the average performance vector. The elements of the vector M represent the average performances of individual systems across the set of queries.

We define the index set $\Phi = \{j_1, ..., j_m\}$ to be a subset of $\{1, 2, ..., n\}$ with $1 \leq m \leq n$ and Q_Φ be the corresponding query subset. We define $M_\Phi \in R^{l\times 1}$ as the column vector comprising the average performance of systems for the subset of queries, Q_Φ. Following the definition in Section 4.3 the aim of a query selection method is to find a subset of queries of a particular size, m, such that the Pearson Linear correlation, ρ_Φ, between the vectors M_Φ and M is maximized.

$$\rho_\Phi = \frac{cov(M, M_\Phi)}{\{var(M)var(M_\Phi)\}^{\frac{1}{2}}} \tag{6.1}$$

such that

$$\begin{aligned} var(M) = &\quad n^{-2}e^T\Sigma e \\ var(M_\Phi) = &\quad m^{-2}d^T\Sigma d \\ cov(M, M_\Phi) = &\quad n^{-1}m^{-1}d^T\Sigma e \end{aligned}$$

where $e = \{1\}^{n\times 1}$ is the vector of n components, each equal to 1; $d \in \{0, 1\}^{n\times 1}$ is a binary vector such that $d_j = 1$ if $j \in \Phi$, and $d_j = 0$ otherwise.

In addition, $\Sigma = cov(X)$ is the $n \times n$ covariance matrix of the system-query performance scores. The $(ij)^{th}$ element of Σ is the covariance between the i^{th} and j^{th} columns of matrix X. The optimum subset maximizes the Pearson correlation ρ_Φ, where, substituting for the variances and covariance, we have

$$\rho_\Phi = \frac{(d^T\Sigma e)}{\{(e^T\Sigma e)(d^T\Sigma d)\}^{\frac{1}{2}}} \tag{6.2}$$

This derivation assumes that the elements of the X matrix are true representatives of system performance which is computed over the full set of relevance judgments. Of course, in practice this assumption does not hold because of the absence of relevance judgments during query selection. In the following section, we propose an extended model that uses performance predictors for approximating systems' true performance. We then extend the model to incorporate explicitly the noise in measurement of system performance.

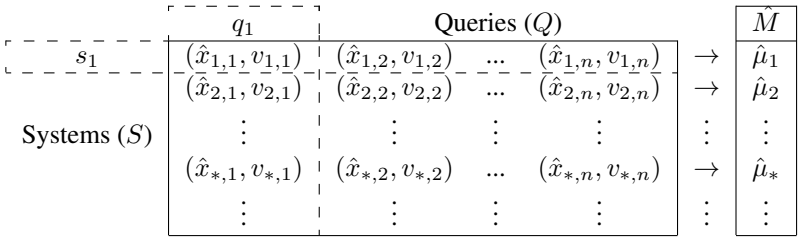


Figure 6.2: The approximated performance matrix $\hat{X}$, for a set of systems and a set of queries. Each pair indicates the estimated performance and associated uncertainty.

6.3 Modeling Uncertainty in Query Selection

We assume that instead of containing the true performance values, each element, x_{sq}, of X holds a predicted performance estimate with a variance from the true value, to which we refer as *uncertainty*. We shortly explain in Section 6.4 that how the predicted performance could be calculated in practice. Hence, the noisy $\hat{X}$ matrix can be represented as shown in Figure 6.2 where each of its elements represents a pair of values: $\hat{x}_{sq}$ and $v_{sq} = var(x_{sq})$. In addition, let $\hat{M}_\Phi \in R^{l \times 1}$ be the vector of l average performance scores computed based on the query subset, Q_Φ, and the performance matrix $\hat{X}$. Thus, in practice we look for a subset that maximizes the Pearson correlation between $\hat{M}_\Phi$ and M. To compute the Pearson correlation we need to compute the variances and covariance of $\hat{M}_\Phi$ and M.

The variance of $\hat{M}_\Phi$ is due to two sources (*i*) the variance across systems, and (*ii*) the variance due to measurement noise. The first variance is expressed by $var(M_\Phi)$ as calculated in Section 6.2. To compute the second variance first note that each of the elements in $\hat{M}_\Phi$ has its own variance. If $\hat{\mu}^i_\Phi$ denotes the performance of i^{th} system in $\hat{M}_\Phi$, then the associated variance is

$$var(\hat{\mu}^i_\Phi) = m^{-2} \sum_{j \in \Phi} v_{ij}$$

Following the law of total variance [Bil95], the variance of $\hat{M}_\Phi$ is given by

$$var(\hat{M}_\Phi) = var(M_\Phi) + E_s(var(\hat{\mu}^s_\Phi)) = \tag{6.3}$$
$$m^{-2} d^T \Sigma d + m^{-2} \sum_{j \in \Phi} E(v_j) = m^{-2} d^T (\Sigma + U) d$$

where $1 \leq s \leq l$ and $U = diag\big(E(v_1), ..., E(v_n)\big)$ is a diagonal matrix, referred to as the uncertainty matrix, also $E(v_q) = l^{-1} \sum_{i=1}^{l} var(x_{iq})$ is the average uncertainty for query q.

To compute the covariance between $\hat{M}_\Phi$ and M, let us consider an unknown system that is randomly sampled, and let x and $\hat{x}$ denote the associated performance row vectors in X and $\hat{X}$. The system's average performance computed based on X and the full set of queries is

$$\mu = n^{-1} xe$$

Also the systems' average performance based on the subset of m queries, Q_Φ, and $\hat{X}$ is

$$\hat{\mu}_\Phi = m^{-1}\hat{x}d$$

where $e \in \{1\}^{n\times 1}$ and $d \in \{0 \text{ or } 1\}^{n\times 1}$ are the column vectors as defined in Section 6.2. The covariance between $\hat{M}_\Phi$ and M is then

$$\begin{aligned} cov(\hat{M}_\Phi, M) \equiv cov(\hat{\mu}_\Phi, \mu) = m^{-1}n^{-1}cov(\hat{x}d, xe) = \\ m^{-1}n^{-1}d^T cov(\hat{x}^T, x)e = m^{-1}n^{-1}d^T\Sigma e \end{aligned} \tag{6.4}$$

where $\hat{x}d = d^T\hat{x}^T$, and

$$\begin{aligned} cov(\hat{x}^T, x) = cov(x^T + \epsilon, x) = \\ E\big\{\big(x - E(x)\big)^T\big(x - E(x)\big)\big\} \equiv cov(X) = \Sigma \end{aligned}$$

Note that, $\hat{x}^T = x^T + \epsilon$ where $\epsilon \in R^{1\times n}$ is the vector of estimator's noise.

Thus, the Pearson correlation between $\hat{M}_\Phi$ and M is given by

$$\hat{\rho}_\Phi = \frac{(d^T\Sigma e)}{\big\{\big(e^T\Sigma e\big)\big(d^T(\Sigma + U)d\big)\big\}^{\frac{1}{2}}} \tag{6.5}$$

Formally, we seek a subset Q_Φ that maximizes $\hat{\rho}_\Phi$. Reordering the correlation above we have

$$\gamma_\Phi \equiv (e^T\Sigma e)^{\frac{1}{2}}\hat{\rho}_\Phi = \frac{(e^T\Sigma d)}{\big(d^T(\Sigma + U)d\big)^{\frac{1}{2}}}$$

Selecting queries for the subset Φ that maximizes $\hat{\rho}_\Phi$ is equivalent to selecting a set of queries that maximizes γ_Φ since $(e^T\Sigma e)^{\frac{1}{2}}$ is a constant. Let σ_{ij} be the $(i,j)^{th}$ element of Σ and $E(v_j)$ be the j^{th} diagonal element of the uncertainty matrix U. Thus we can rewrite γ_Φ as

$$\max_\Phi \gamma_\Phi = \frac{\sum_{1\le i\le n, j\in\Phi}(\sigma_{ij})}{\big\{\sum_{i,j\in\Phi}(\sigma_{ij}) + \sum_{j\in\Phi}E(v_j)\big\}^{\frac{1}{2}}} \tag{6.6}$$

Equation 6.6 provides valuable insight into the query selection problem. In order to maximize γ_Φ we aim at a set of queries that minimizes the denominator and maximizes the numerator.

To minimize the denominator, we should choose the m queries that are least correlated with one another. This is equivalent to maximizing the information we derive from each query in the subset. Conversely, if the columns of $\hat{X}$ are perfectly correlated, then all the queries provide the same information and we may as well have a subset of size one. Additionally, the sum of the expected variances, $E(v_j)$, of the selected queries should be a minimum.

The numerator is maximized if the subset of query-systems has high correlation with the rest of the queries. This is also intuitively clear. After all, if the subset of query-systems is completely uncorrelated

with the remaining query-samples, then this subset can provide no prediction of how systems will perform on the remaining queries. Assuming that an evaluation on the full set of query-systems vectors is a golden standard, the objective encodes a preference for subsets that have a strong correlation with the full set of queries. In the next section, we describe how the theoretical uncertainty-aware model introduced in this section can be applied in practice.

6.4 Adaptive Query Selection

So far, we introduced an uncertainty-aware query selection model that extended previous work by explicitly modeling uncertainty and allowing the elements of query-system matrix to be replaced with predicted performance, rather than the actual performance values. Equation 6.6 shows how predicted performance values can be incorporated in the optimization process, but does not indicate how they can be computed in practice. In this section, we propose an *adaptive* method that iteratively selects queries and refines the estimations in $\hat{X}$. This method exploits supervised prediction and uses the relevance judgments of queries selected already, to train a model for selecting subsequent queries.

Our adaptive method iteratively selects a query, collects its associated relevance judgments, and uses the relevance judgments of queries that are selected so far to predict the relevance judgments of non-selected queries. It subsequently estimates the associated system-query performance scores and produces the corresponding uncertainty, and updates the $\hat{X}$ matrix by adding the systems' performance scores measured for the selected query, and those predicted for the non-selected queries. We repeat this until we reach the maximum number of queries to be selected.

At each iteration, in order to predict the relevance of documents for queries that have not been selected, we train a classifier using judged documents of previously selected queries as training data. Each query-document pair is represented to the classifier as a vector of 7+l generic features where l refers to the number of systems. These features are:

- The number of systems that retrieved the query-document pair (one feature).
- The average, minimum and maximum ranks given to the query-document pair by systems (three features).
- For systems that retrieve the query-document pair, we calculate their corresponding past-performance scores based on the subset of queries for which we have relevance judgments. For example, if the metric is AP, we compute a system's MAP based on its AP scores obtained for previously selected queries. We then determine the minimum, maximum and average across systems (three features).
- The l relevance scores provided by l systems for the given query-document pair (l features). If a system does not retrieve the document, the corresponding score is set to the minimum of the scores of the other documents retrieved by that system.

We use a linear support vector machine (SVM) [CV95] as our classifier. For each query-document pair, we then map the output of the classifier to a probability score using the calibration method proposed

in [Pla00]. Briefly, let $f \in [a, b]$ be the output of classifier. We use a sigmoid function to map f to a posterior probability on [0,1]:

$$p_i = P(r_i = 1|f_i) = \frac{1}{1 + exp(Af_i + C)}$$

where r_i refers to the true relevance value of document i, p_i is its probability of relevance, and A and C are the parameters of sigmoid function that are fitted using maximum likelihood estimation from a calibration set (r_i, f_i). The training data is the same as the training data used by the classifier. Thus, at each iteration we retrain the classifier and fit the sigmoid function to exploit the increase in training data from the new round of relevance judgments.

After each query-document pair is assigned a probability of relevance, we use these probabilities in the family of estimators, referred to as MTC, proposed by Carterette et al. [CAS06] to provide new estimates for the unknown values in the $\hat{X}$ matrix. For example, when the metric of interest is $P@k$, the expectation and variance are calculated as:

$$\begin{aligned} E[P@k] = &\quad \tfrac{1}{k} \textstyle\sum_{i=1}^{k} p_i \\ var[P@k] = &\quad \tfrac{1}{k^2} \textstyle\sum_{i=1}^{k} p_i(1 - p_i) \end{aligned}$$

where p_i is the calibrated relevance probability of the document retrieved at rank i. The formulations of other metrics, e.g. AP, can be found in [CAS06].

6.5 Evaluation Settings

Query selection methods are often evaluated according to the ranking they produce for systems, compared with the ground-truth ranking that is computed based on all the queries and the full set of associated relevance judgments. As in most previous work in this area, such as [GMR09, Rob11], we also use Kendall-τ and Pearson coefficient as our correlation metrics. Kendall-τ penalizes disordering of high-performance and low-performance system pairs equally. However, in practice, distinguishing between best performing systems is often more important. Therefore, we also report separate results specifically on subsets of best performing systems in many of our experiments. We report separate results for average precision (AP) and precision at position 100 ($P@100$) as our system performance metric.

We run our experiments on (*i*) TREC 2004 Robust track comprising of 249 queries, 110 runs and 311,410 relevance judgments, and (*ii*) TREC-8 Ad-hoc track comprising of 50 queries, 129 runs and 86,830 relevance judgments. In our experiments, we consider runs as search systems, taking special care when considering runs from the same site. We also create a web test collection based on the query logs of a commercial search engine. This dataset comprises 1,000 queries, 50 runs of a learning to rank system [Liu09] trained with different feature sets, and 30,000 relevance judgments.

We compare the performance of our query selection method against three baselines, namely: random, oracle and *IQP*.

Random: randomly selects a subset of queries. We report the results averaged over 10,000 random

trials and consider 95% confidence interval of the sample average.

Oracle: the associated results are provided with the full X matrix constructed from the full set of queries and all the relevance judgments in the associated test collection. For a given subset size $m < 10$ and $m > (n - 10)$, we perform an exhaustive search to find the oracle subset. Exhaustive search is computationally expensive for $10 < m < (n - 10)$. Therefore, we estimate the best subset of size $10 < m < (n - 10)$ by randomly generating 10,000 query subsets from which the best subset is selected.

Iterative Query Prioritization (IQP): to investigate the effect of incorporating uncertainty in query selection we also consider a modified version of our query selection model in which uncertainty in measurement is ignored. We call it iterative query prioritization (*IQP*). Similar to our adaptive query selection *IQP* starts from zero relevance judgments and iteratively selects queries. However, *IQP* does not consider the uncertainty in estimating the entries of the *X* matrix. Therefore, the elements of the corresponding uncertainty matrix U is 0, and consequently omitted from the optimization in Equation 6.6. That is, *IQP* uses the same classifier, as in our adaptive method, but directly maps the output of the classifier to 0 or 1, when the relevance judgments are binary, and regards them as the predicted absolute relevance values. Therefore there is no calibration of relevance probabilities involved. As such, it does not use the MTC estimators discussed in Section 6.4 and, instead, uses standard metrics, e.g. AP, to measure systems' performance.

6.6 Experimental Results

In the experiments with TREC test collections, we considered all the official retrieval runs. Each system contributed 100 documents to the pool for each query. After selecting a query, the official TREC judgments were collected and revealed. The Adaptive and *IQP* methods, then added these recently judged documents to their training sets.

On each test collection, we report the results for three different groupings of systems: (*i*) all systems, (*ii*) top 30 best performing systems, and (*iii*) only pairs of systems with a statistically significant performance difference, measured by the paired t-test at significance level 0.05.

Figure 6.3 shows the results on the Robust 2004 test collection with 249 queries. The systems evaluation metric was AP. Pearson linear correlation was used to measure the correlation of a query subset vector $\hat{M}_\Phi$, and corresponding vector M, calculated using the full set of 249 queries. At the initialization step of the Adaptive and IQP methods, the first query was randomly selected. To deal with the variation of random sampling, we considered 50 trials. In each trial, we randomly selected the first query and then ran the Adaptive and IQP methods to select subsequent queries. This process was repeated 50 times, each time a new query was selected as the first choice. The average of 50 trials was then reported as the average results for Adaptive and IQP. We also considered the 95% confidence interval of the average performance to detect significant differences between the query selection methods. For instance, when the subset covered 28% of the full query set, the average of the 50 Pearson correlation scores obtained by the Adaptive method in 50 trails was 0.94 and the associated standard deviation was 0.07. Thus the 95% confidence interval was: $\left[0.94 \pm 1.96 \times \frac{0.07}{\sqrt{50}}\right]$. The confidence intervals are shown

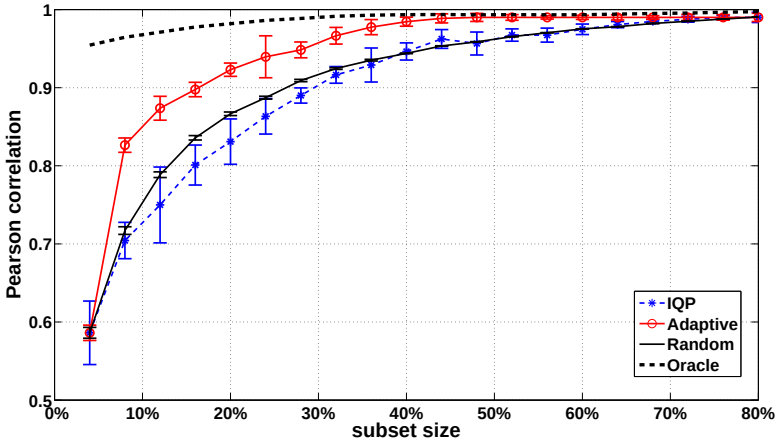


Figure 6.3: The Pearson linear correlation between M and M_Φ. The query subsets are selected using (*i*) Oracle, (*ii*) random, (*iii*) *IQP*, (*iv*) Adaptive query selection algorithm, for the Robust 2004 test collections with 249 queries. The first query is randomly selected. The results are averaged over 50 trials with AP metric.

as error bars in Figure 6.3. In general, the difference between two methods is statistically significant at a particular subset size, if the associated error bars do not overlap.

We also repeated the experiments with Kendall-τ rank correlation. Figure 6.4 shows the associated results. As seen, in Figure 6.3 and 6.4, for both Pearson correlation and Kendall-τ, the Adaptive method significantly outperformed the Random and *IQP* baselines across different subset sizes. As in Figure 6.4 the Adaptive method achieved a Kendall-τ correlation of 0.9 with a subset that covers 50% of the queries (125 out of 249 queries). However, the Random and *IQP* methods required at least 70% of queries to achieve the same Kendall-τ. Surprisingly, *IQP* performs no better than Random, and for initial subsets it even performs worse than Random. This is because *IQP* relies on the predicted performance scores and ignores uncertainty in estimations that may lead to the selection of inefficient queries.

Table 6.1 summarizes the Kendall-τ and Pearson correlation of the four different query selection methods obtained for selecting $\{20, 40, 60\}\%$ of queries in Robust 2004 and TREC-8 test collections.

The columns labeled 'all' indicates the results of considering all the systems in a test collection when measuring Pearson and Kendall-τ correlations. For both test collections and all subset sizes, $\{20, 40, 60\}\%$, the Adaptive method significantly outperformed *IQP* and Random baselines in most cases. The significance differences, marked by †, were calculated the same way as in Figure 6.3. For instance, in the Robust test collection the adaptive method obtained $\{15, 10, 5\}\%$ improvements, on average, in Kendall-τ correlations over Random and *IQP* for subsets of $\{20, 40, 60\}\%$ respectively. Similar improvements were observed for the TREC-8 test collection.

The columns labeled 'top' indicates the results for considering only the top 30 best performing systems, i.e. those that obtained the highest MAP scores in the original test collection. We do this experiment to follow a common trend in IR experiments in which the precise estimate of top performing systems is only of interest. When calculating Pearson and Kendall-τ correlations, the vectors $\hat{M}_\Phi$ and M were constructed only based on the top 30 systems. Here, the remaining systems only contributed to the

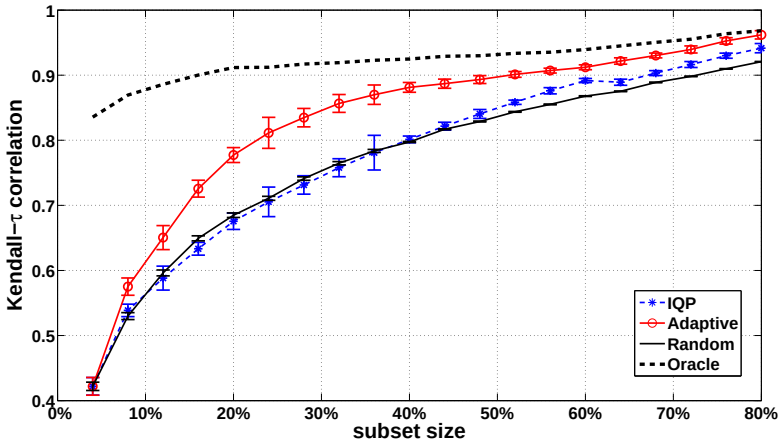


Figure 6.4: The Kendall-τ linear correlation between M and M_Φ. The query subsets are selected using (*i*) Oracle, (*ii*) random, (*iii*) *IQP*, (*iv*) Adaptive query selection algorithm, for the Robust 2004 test collections with 249 queries. The first query is randomly selected. The results are averaged over 50 trials with AP metric.

Table 6.1: Comparisons of four query selection methods based on the AP metric and two TREC test collections. The statistically significant improvements of *Adaptive* over *IQP* and Random are marked by †.

Subset	**Method**	**Robust2004**					**TREC-8**				
		Kendall-τ			**Pearson**		**Kendall-τ**			**Pearson**	
		all	**top**	**sig**	**all**	**top**	**all**	**top**	**sig**	**all**	**top**
20%	**Random**	0.68	0.45	0.75	0.83	0.68	0.72	0.45	0.88	0.92	0.77
	IQP	0.67	0.47	0.78	0.86	0.70	0.74	0.53	0.92	0.93	0.81
	Adaptive	**0.77**†	**0.63**†	**0.85**†	**0.92**†	**0.79**†	**0.83**†	**0.69**†	**0.95**†	**0.95**†	**0.92**†
	Oracle	0.90	0.81	0.90	0.97	0.95	0.88	0.80	0.97	0.97	0.95
40%	**Random**	0.80	0.58	0.82	0.93	0.76	0.77	0.58	0.95	0.95	0.86
	IQP	0.80	0.56	0.85	0.94	0.78	0.81	0.66	0.96	0.95	0.89
	Adaptive	**0.87**†	**0.69**†	**0.89**†	**0.98**†	**0.89**†	**0.90**†	**0.81**†	**0.99**†	**0.97**†	**0.95**†
	Oracle	0.92	0.86	0.95	0.99	0.96	0.93	0.85	1.0	0.98	0.97
60%	**Random**	0.85	0.71	0.88	0.97	0.90	0.87	0.70	0.97	0.97	0.90
	IQP	0.88	0.73	0.90	0.96	0.91	0.88	0.80	0.99	0.98	0.92
	Adaptive	**0.91**†	**0.83**†	**0.95**†	**0.99**†	**0.96**†	**0.93**†	**0.85**†	1.0	0.98	**0.96**†
	Oracle	0.94	0.92	0.97	0.99	0.99	0.95	0.91	1.0	0.99	0.99

Table 6.2: Comparisons of four query selection methods based on the *P@100* metric and two TREC test collections. The statistically significant improvements of *Adaptive* over *IQP* and Random are marked by †.

Subset	Method	Robust2004					TREC-8				
		Kendall-τ			Pearson		Kendall-τ			Pearson	
		all	top	sig	all	top	all	top	sig	all	top
20%	**Random**	0.64	0.43	0.69	0.78	0.67	0.67	0.43	0.86	0.88	0.74
	IQP	0.65	0.45	0.74	0.84	0.69	0.76	0.50	0.90	0.91	0.80
	Adaptive	**0.73**†	**0.60**†	**0.79**†	**0.89**†	**0.75**†	**0.80**†	**0.68**†	**0.93**†	**0.95**†	**0.91**†
	Oracle	0.88	0.80	0.91	0.98	0.94	0.86	0.82	0.99	0.99	0.94
40%	**Random**	0.78	0.55	0.81	0.90	0.77	0.74	0.56	0.92	0.95	0.82
	IQP	0.81	0.53	0.82	0.90	0.77	0.80	0.61	0.95	0.93	0.91
	Adaptive	**0.86**†	**0.70**†	**0.87**†	**0.96**†	**0.89**†	**0.91**†	**0.80**†	**0.98**†	**0.98**†	**0.96**†
	Oracle	0.92	0.85	0.94	0.98	0.95	0.94	0.83	0.99	0.99	0.97
60%	**Random**	0.84	0.73	0.87	0.95	0.91	0.85	0.72	0.98	0.94	0.91
	IQP	0.86	0.74	0.89	0.94	0.92	0.87	0.81	0.99	0.98	0.93
	Adaptive	**0.90**†	**0.81**†	**0.93**†	**0.97**†	**0.95**†	**0.94**†	0,82	1.0	0.97	0.94
	Oracle	0.93	0.92	0.96	0.97	0.99	0.96	0.92	1.0	1.0	0.98

query selection process and were not used for evaluation. Once again, the Adaptive method significantly outperformed the *IQP* and Random methods in most of the cases. Interestingly, the improvements were even larger than the improvements obtained when evaluating the full set of systems. For instance, for the Robust test collection, when evaluating the full set of systems the improvement in Kendall-τ was 10% on average. However, when considering only top performing systems the average improvement rose to 25%. Similarly, the average improvement in Pearson correlation rose from 7% to 14% on average. Similar results were observed for TREC-8 test collection.

The columns labeled 'sig' indicates the results when only considering pairs of systems whose performances difference is statistically significant. When a difference in average performance scores of two systems is not statistically significant, it is reasonable that they may be ordered differently when evaluated over a subset of queries. Such tied systems increase the probability of a swap in ordering systems and may considerably decrease Kendall-τ. This is because the common formulation of Kendall-τ, which is also used in our experiments, is not able to distinguish between pairs of systems with and without significant differences. This is the case for the Robust and TREC-8 test collection where about 30% of pairs of systems are tied, measured by paired t-test at significance level 0.05. Thus, we also measured the Kendall-τ value obtained by the four query selection methods when only evaluating pairs of systems with a significant difference in MAP. Again, the Adaptive method significantly outperformed IQP and Random in most cases.

We repeated the experiments for $P@100$ metric, and observed similar results for both the test collections. The associated result is summarized in Table 6.2.

6.6.1 Results of the Web Data

We also investigated the performance of the Adaptive method on a test collection comprising web search results from a commercial search engine with 1,000 queries and 50 systems (see Appendix C). Various rankers (runs) of a learning to rank system that were trained with different feature sets were considered

Table 6.3: Comparisons of the random and adaptive methods using a web test collection of a commercial search engine.

	Method	desired Kendall-τ		
		0.7	**0.8**	**0.9**
#queries	**Random**	167	368	739
	Adaptive	71	207	486
#relevance judgments	**Random**	5010	10235	28804
	Adaptive	2086	5803	15854

as participating systems. To generate a ranker we randomly sampled $g = \{5, 10, 20, 30 \text{ or } 40\}$ features from a given feature set and optimized the ranker on a common training set. For each query, the top 5 web pages returned by the rankers were pooled for relevance assessment. The performance of each ranker was measured according to precision at position 5 ($P@5$).

Table 6.3 reports (i) the number of queries, and (ii) the number of relevance judgments required to reach Kendall-τ values of $\{0.7, 0.8, \text{and } 0.9\}$ by (1) Adaptive, and (2) Random query selection method. We focused on the comparisons between the Adaptive and Random. That was because the results obtained by the *IQP* method was no better than the results of random sampling. Also, since the random sampling is the common method used in IR community to select a set of queries, the comparison between Adaptive and Random provided estimates of the cost reduction caused by the adaptive method in practice.

The results of the Adaptive method are the average of 10 trials. In each trial, at the initialization step of the Adaptive method we selected a sample of 20 queries instead of randomly selecting one query. This ensured a sufficiently large training set for the classifier at the first step without losing much efficiency in the query selection performance.

As seen in Table 6.3, the required subset sizes for τ=$\{0.7, 0.8, 0.9\}$ are statistically significantly smaller than those required for random sampling. For instance, the random method obtains $\tau = 0.9$ by a subset of size 739 whereas the Adaptive method only requires 486 queries to reach the same τ. This is equivalent to judging 12950 fewer documents than those required by the random method, and the associated cost is correspondingly reduced. Similar results are observed for τ=$\{0.7 \text{ and } 0.8\}$.

6.6.2 Effects of Initialization

In the previous experiments, we randomly selected the first query at the initialization step. We now consider the sensitivity of the Adaptive method to the selection of the first query. The choice of the first query could possibly affect both (*i*) the quality of the queries selected in the subsequent stages and (*ii*) the training data for the classifier. Our analysis only focuses on (*i*) as the effects on (*ii*) highly depend on the classification method which is out of the scope of our work.

In order to isolate the effects on the quality of the subsequent queries, we assumed that the true matrix X was available, i.e., that the estimator had access to all relevance judgments for computing the true performance values in X. Using the TREC-8 data set, we randomly selected the first query. Subsequent queries were iteratively selected based on the query selection model in Equation 6.6 but using the true matrix X where the corresponding uncertainty matrix U was zero. Results are shown in

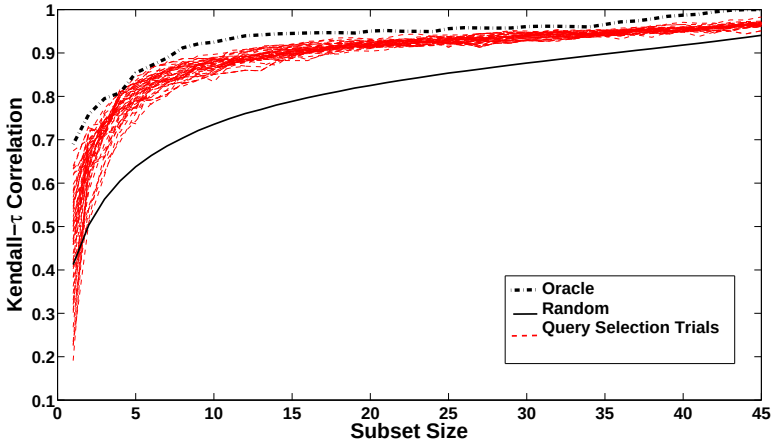


Figure 6.5: Sensitivity of the query selection to the first query using TREC-8 comprising 50 queries. The subset size varies between 1 and 45.

Figure 6.5 for 50 trials. Each trails contained a distinct query for the initialization. As seen, the subsets from all the trials converged to the oracle's result more quickly than the average of Random sampling. The kendall-τ variation across trials decreased as more queries were selected. For the subset sizes greater than 10 queries, performance was very similar across all the trials. This suggests that the query selection model is robust to the selection of the first query. Thus, no matter what query is selected at the first step, the subset chosen by the method quickly converges to the optimal subset if the estimator is noise free.

6.7 Generalization

We consider the generalizability problem of our query selection method. In section 6.7.1 we discuss the generalizability of a query subset in terms of reliably evaluating a set of new systems that do not contribute to the query selection process. As such a query subset is known generalizable if it leads to reliable evaluation results of different sets of systems. In Section 6.7.2 we discuss the generalizability of a query subset across multiple metrics. This is indeed important when query subsets are used to evaluate systems by various metrics.

6.7.1 Evaluation of New Systems

Previous work [Rob11, HCMF$^+$11] showed that queries selected by a particular set of systems may not be able to provide reliable conclusions when used to evaluate a set of new previously unseen systems. We also observed this with our Adaptive algorithm.

To avoid over-fitting the query subset to the systems used to select the queries we modify the Adaptive algorithm. The modified version is referred to as 'Adaptive$^+$'. When selecting a query we consider $c(c > 1)$ random subsets of the l systems of size $h(h < l)$. We allow overlaps between the subsets and ensure that each system appears in at least one of the subsets. For each subset of systems we choose a query that, in combination with already selected queries in Φ, maximizes γ_Φ. Finally, we pick the query that is selected by most of the subsets of systems, and consider it for the next round of relevance judgments.

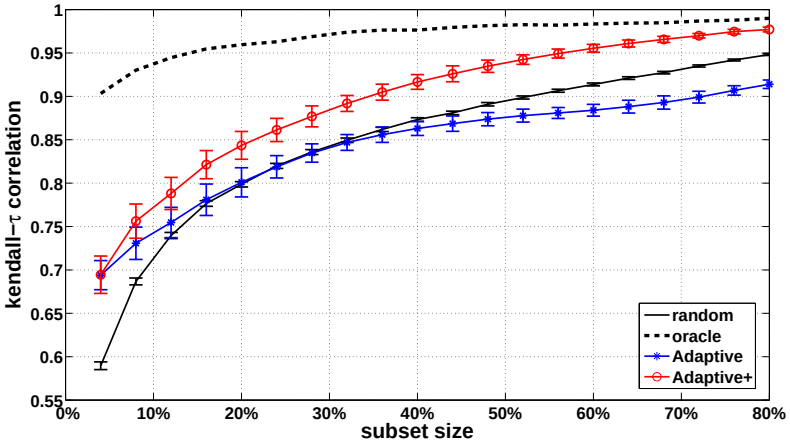


Figure 6.6: The generalizability test for query subsets selected by (*i*) Adaptive: our query selection method 'without' a generalizability module, (*ii*) Adaptive$^+$: our query selection method with a generalizability module.

We tested the generalizability of a query subset using the two TREC test collections: TREC 2004 Robust and TREC-8 Ad-Hoc tracks. We first randomly selected 50% of systems in a TREC test collection and treated them as new systems. The rest of systems were considered as participating. When selecting new systems, we held out not only individual runs but the entire set of runs from the same participant (site). Furthermore, during the computation of performance metrics for participating systems, we removed documents that were uniquely retrieved by the new (held-out) systems. We then used participating systems to select queries by a query selection method, and then assessed the generalizability of the selected subset based on the evaluation of new systems.

The results of the generalizability test using the Robust test collection and Kendall-τ is shown in Figure 6.6. We created $c = 100$ random subsets, each of size $h = 0.2 \times l$, where l refers to the number of participating systems. Figure 6.6 clearly shows that the Adaptive algorithm performs no better and sometimes worse than random. This is because of over-fitting to the participating systems. In contrast, the Adaptive$^+$ significantly outperforms the random sampling across the different subset sizes. The significant differences are calculated as explained in Table 6.1.

The detailed results of generalizability experiments are shown in Table 6.4 for two test collections, Robust and TREC-8, and two metrics, AP and $P@100$. In all cases the Kendall-τ obtained by Adaptive$^+$ is significantly larger than the Kendall-τ of the Adaptive and Random algorithms.

6.7.2 Use of Alternative Performance Metrics

One of the goals of IR test collections is to enable the evaluation of systems in terms of various metrics. As a result, the set of queries that are used for evaluation must be able to provide precise estimates of systems' performance for various metrics. In the following experiments, we show when the metric used to select a subset of queries differ the metric used for evaluation, the query subset may not provide a reliable summary of systems evaluation.

We modify the Adaptive algorithm to select a subset that is suitable for a set of metrics. The

Table 6.4: Comparing the generalizability of a selected subset using two metrics: $P@100$ and AP. Statistically significant differences are indicated by †.

Subset	Method	kendall-τ			
		Robust 2004		TREC-8	
		P@100	AP	P@100	AP
20%	**Random**	0.77	0.80	0.75	0.78
	Adaptive	0.76	0.82	0.76	0.80
	Adaptive$^+$	**0.84**†	**0.87**†	**0.81**†	**0.85**†
	Oracle	0.91	0.95	0.86	0.89
40%	**Random**	0.82	0.87	0.84	0.85
	Adaptive	0.80	0.85	0.84	0.86
	Adaptive$^+$	**0.89**†	**0.92**†	**0.90**†	**0.90**†
	Oracle	0.93	0.97	0.94	0.93
60%	**Random**	0.89	0.91	0.89	0.90
	Adaptive	0.84	0.88	0.87	0.91
	Adaptive$^+$	**0.93**†	**0.96**†	**0.95**†	**0.95**†
	Oracle	0.96	0.98	0.97	0.97

modified version is referred to as 'Adaptive*'. At each step of the query selection process, for each of the metrics and each of the non-selected queries the Adaptive* computes the associated γ_Φ scores. It then computes the average of a set of γ_Φ scores that a query obtains across the metrics. Finally it selects the candidate query with the maximum *average* of γ_Φ scores. Thus, before selecting a query we consider the γ_Φ scores it obtains across the metrics, and select a query with the maximum average of γ_Φ scores.

We consider four IR metrics: $P@10$, $P@100$, $Recall$ and AP. We use each of the metrics and select query subsets of various sizes and measure the associated Kendall-τ scores (T_1). Also let T_2 be the set of Kendall-τ scores for various subset sizes calculated when the metric used for measuring systems performance (evaluation metric) is different from the metric used for query selection (selection metric). Ideally we would like the Kendall-τ scores in T_2 not to be considerably smaller than those in T_1. To measure the distances between T_1 and T_2 scores we measure $\big(mean(T_2) - mean(T_1)\big)$ as the average loss Kendall-τ.

Table 6.5 represents the results of our experiment using the Robust 2004 test collection. Each of the four metrics were used both as selection metric, to form $\hat{X}$ matrix and select a query subset, and evaluation metric, to measure a system's performance. For instance, the average Kendall-τ loss scores of the four evaluation metrics are shown in the first row when $P@10$ is used as the selection metric to choose the query subsets. Clearly, when the selection metric and the evaluation metric are the same, the average loss is 0.

As seen in Table 6.5, when the Recall is the selection metric, the average loss of $P@10$ and $P@100$ are minimum. The average loss of AP, as an evaluation metric, is minimum when $P@10$ is the selection metric. Also, when $P@100$ is the selection metric, the average loss of Recall is minimum. However, there is not a unique selection metric that results a minimum loss for other metrics.

We also selected queries by using the Adaptive* method. As seen when using the Adaptive* method, the average loss for all the metrics was considerably reduced. The last row of Table 6.5 also represents the results of random sampling averaged over 1000 trials. To investigate whether the subsets selected by a metric significantly outperform the random subsets the statistical significant differences in

Table 6.5: The average Kendall-τ loss $\big(mean(T_2)$ - $mean(T_1)\big)$ for four various metrics using TREC 2004 Robust track. Given a metric α, T_1 denote the set of Kendall-τ scores across various subset sizes obtained when the metric α is used for both query selection and system evaluation; T_2 denote the set of Kendall-τ scores obtained when the metric α is used to measure systems performance on a subset of queries that is selected by another metric.

Selection Metric	**Evaluation Metric**			
	P@10	**P@100**	**Recall**	**AP**
P@10	0.0	-0.082	-0.065	-0.051
P@100	-0.084	0.0	-0.042	-0.068
Recall	-0.076	-0.063	0.0	-0.073
AP	-0.089	-0.070	-0.062	0.0
Adaptive*	**-0.011**†	**-0.012**†	**-0.018**†	**-0.014**†
Random	-0.114	-0.086	-0.056	-0.078

average Kendall-τ loss were measured using the paired t-test at significant level 0.05. Table 6.5 shows that subsets selected by the Adaptive* lead to average Kendall-τ losses that are significantly smaller than the average loss obtained by the random subset.

6.8 Summary

We assumed there is a set of compiled queries by which we intend to evaluate systems. However, budget constraints only permitted collecting relevance judgments for a subset of queries. Thus, our goal was to select a representative subset of queries that provided a close approximation of systems' performance computed when using the full set of queries. We provided a mathematical model for selecting queries. Our model explicitly formulated the uncertainty in performance scores that were introduced by the absence of relevance judgments. The mathematical formulation showed that the optimal subset of queries should be least correlated with each other but have the maximum correlation with the rest of queries. Also, the total uncertainty associated with selected queries should be minimum.

We proposed an Adaptive algorithm in which queries were iteratively selected and relevance judgments were obtained for each query immediately after it was added to the subset. These relevance judgments were then used by a classifier to aid the selection of subsequent queries. Of course, in practice, the result of the Adaptive method is sensitive to the accuracy of classifier. We used the SVM classifier, that is reported as one of the strong text classifiers [CV95], in our experiment. We demonstrated the effectiveness of our Adaptive algorithm using two TREC test collections and a web test collection of a commercial search engine. For all the three test collections, the Adaptive algorithm significantly outperformed the existing baselines.

Query subset selection methods have been criticized for not usually generalizing to previously unseen systems. Our Adaptive algorithm also exhibited this problem. However, we refined the algorithm and showed that the extended algorithm does indeed generalize to new systems. We also modified the Adaptive algorithm to select queries across multiple evaluation metrics. Our experiments on TREC data demonstrated the ability of the algorithm to find a global subset that leads to reliable evaluation for various metrics.

Chapter 7

Unified Budget Allocation

We consider the problem of optimally allocating a fixed budget to construct relevance judgments for an information retrieval test collection, such that it can (*i*) accurately evaluate the relative performance of the participating systems, and (*ii*) generalize to new, previously unseen systems. We address this problem by integrating the query selection and the document selection approaches to form a unified budget allocation approach.

The budget allocation is formulated as a convex optimization problem, thereby providing a flexible framework to incorporate various constraints. We introduce a generalizability constraint and show how it can increase the effectiveness of the test collection for comparative evaluation of new systems.

We devise an iterative algorithm to implement the budget allocation model in practice. Our iterative algorithm apportions the budget between several steps. At each step, all the query-document pairs are evaluated and a portion of budget is optimally allocated across a set of query-document pairs with the highest priority scores. The associated relevance assessments are then acquired and used to aid the allocation of budget in the next step.

We evaluate our unified budget allocation approach on two TREC test collections namely TREC-8 Ad-hoc track and TREC 2004 Robust track. We demonstrate that our allocation method is cost efficient and yields a significant improvement in the generalization of the test collections.

7.1 Introduction

An IR test collection is typically constructed in conjunction with a set of participating IR systems. Each participating system retrieves a set of documents in response to each test query and these sets are pooled together. Relevance judgments are then obtained only for documents in the pool and specific metrics are used to compare systems performance. While the number of relevance judgments needed is greatly reduced, economic constraints may still prevent exhaustive judgments of all documents in the pool.

We consider how to prioritize query-document pairs for relevance judgments, when budget constraints preclude obtaining relevance judgments for all the pooled documents. We formulate the question as an optimization problem in which, for a given budget, we seek to identify a set of query-document pairs that most accurately evaluate the participating systems and provide the best generalization to yet unseen systems. The latter refers to systems that have not contributed to the pool of evaluated documents.

Our work is, in part, motivated by the recent developments in document selection approaches, e.g. [CAS06, YA06, APY06], that enable accurate estimates of systems' effectiveness at query level by only judging a few number of documents. Furthermore, we showed in Chapter 4 that identifying a small set of representative queries can lead to system evaluations that are equivalent in quality to those based on much larger sets of queries. By connecting these aspects with the need to generalize the IR test collections to new systems and explicitly manage the cost of relevance assessments, we provide a unified budget allocation optimization for optimally collecting a set of relevance judgments.

The main contributions are (*i*) formulation of the budget allocation problem as a convex optimization to provide a flexible framework to incorporate various constraints, (*ii*) the incorporation of a generalization constraint based on the estimated number of unjudged relevant documents, and (iii) the implementation of the convex optimization through incremental acquisition of relevance judgments.

7.2 The Budget Allocation Strategy

Let S denote the population of all IR systems. Although the distribution of S is unknown, we assume that all, past present and future, systems are drawn from this distribution. This is a simplifying assumption but a good starting point for developing the mathematical model.

We are given a document corpus D and a set of n test queries $\{q_1, q_2, ..., q_n\}$. We assume that there is a set of l participating systems ($S_l \subset S$), each of which returns a number of retrieved documents for each of the n queries. From the retrieved documents we create a common pool of documents to be used for comparative evaluation of the systems. Let Ω denote the cost of building relevance judgments over the pooled documents. For a given budget B, that is much smaller than Ω ($B \ll \Omega$), we seek to collect relevance judgments for a subset of query-document pairs in order to accurately evaluate the performance of the participating systems and reliably estimate the performance of yet unseen systems. We divide B in p portions, $\{B_1, ..., B_p\}(p \geq 2)$, and propose an iterative process to allocate the limited budget.

7.2.1 Initialization

In the first iteration, we allocate the first portion of budget, B_1, to assess the relevance of some of the documents in the common pool. Given that there is no prior information about the relevance of documents, the simplest allocation strategy is to divide the budget equally among the n queries and, for each query, select a fixed number of documents to be judged. In the common pooling technique, the documents are ranked based on the query relevance. Thus one can choose a uniform pool depth across queries to select documents to fit the available budget B_1.

7.2.2 Selective Expansion

In iterations between 2 and p, we utilize the associated budget B_k, ($2 \leq k \leq p$), to extend the set of relevance judgments from the previous step. Query-document pairs are prioritized and a subset of them are selected to be judged. The prioritization process is based on a convex optimization of a cost function that seeks to (*i*) achieve maximum agreement with the evaluation of S_l systems using the full set of pooled documents and ideal budget Ω, and (*ii*) generalize to new, unseen systems.

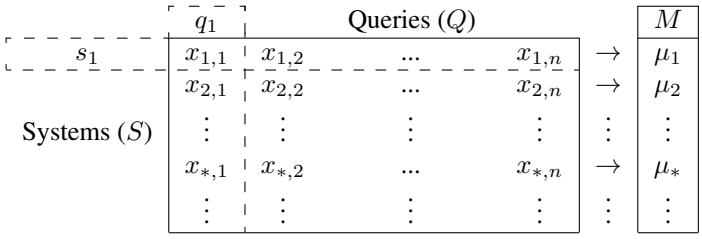


Figure 7.1: The true performance matrix X for a set of system systems and a set of queries. Each entry indicates the system performance score based on the available relevance judgments.

7.3 A Framework for Budget Allocation

When relevance judgments are available for the full set of pooled documents, we observe the retrieval performance of each of the l participating systems over a finite set of n queries. The performance measurements are represented in the form of a performance matrix X as shown in Figure 7.1. Each row corresponds to a system and each column to a query . An entry $x_{s,q}$ in X denotes the performance score, e.g. AP, of the s^{th} system on the q^{th} query. The systems' average performance, e.g. MAP, can also be represented by a column vector $M \in R^{l\times 1}$.

In practice, X matrix and the associated M vector are unobservable because of the absence of relevance judgments for all or some of the pooled documents. Instead, X is approximated by a performance matrix $\hat{X} \in R^{l\times n}$, each element contains a predicted performance estimate, $\hat{x}_{s,q}$, with a variance from the true value, $v_{s,q} = var(x_{s,q})$, referred to as uncertainty (Figure 7.2).[1]

In addition, the average performance of a system can be approximated by the average of the corresponding row vector in $\hat{X}$. In a more general case, the average performance of a system can be expressed as a linear combination of the approximated effectiveness scores, $x_{s,q}$, associating a priority score with each query. We define $\beta \in [0,1]^{n\times 1}$ be a the column vector with real values in [0,1]. Also let $\hat{M} \in R^{l\times 1}$ denote a column vector of systems' average performance which are approximated based on $\hat{X}$ and β:

$$\hat{M} = \hat{X}\beta$$

At each iteration, a portion of budget, $B_k(1 \leq k \leq p)$, is used to expand the set of relevance judgments. Subsequently, the elements of $\hat{X}$ are updated and used to estimate the elements of β vector as priority scores of queries in order to construct relevance judgments in the next iteration.

The goal is to set the priority scores of representative queries to be higher than the other queries. To do so, the value of β coefficients are chosen such that the $\hat{M}$ vector closely approximates M vector. We use Pearson linear correlation, as in Chapter 6, to measure the closeness between $\hat{M}$ and M.

The Pearson linear correlation between $\hat{M}$ and M is given by

$$\rho_\beta = \frac{cov(M,\hat{M})}{\{var(M)var(\hat{M})\}^{\frac{1}{2}}} \tag{7.1}$$

[1]When there is no relevance judgments, we can use a query performance predictor, e.g. [SNC01], to approximate $\hat{X}$ matrix. In addition, when an initial set of relevance judgments is already collected, we can use the metrics designed for partial relevance judgments, e.g. [CAS06, YA06, AP08], to measure $x_{s,q}$ scores and associated variances. This can the be used to assign further relevance judgments given a set of initial judgments.

	q_1	Queries (Q)				$\hat{M}$
s_1	$(\hat{x}_{1,1}, v_{1,1})$	$(\hat{x}_{1,2}, v_{1,2})$	...	$(\hat{x}_{1,n}, v_{1,n})$	$\rightarrow$	$\hat{\mu}_1$
	$(\hat{x}_{2,1}, v_{2,1})$	$(\hat{x}_{2,2}, v_{2,2})$	...	$(\hat{x}_{2,n}, v_{2,n})$	$\rightarrow$	$\hat{\mu}_2$
Systems (S)	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
	$(\hat{x}_{*,1}, v_{*,1})$	$(\hat{x}_{*,2}, v_{*,2})$	...	$(\hat{x}_{*,n}, v_{*,n})$	$\rightarrow$	$\hat{\mu}_*$
	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Figure 7.2: The approximated performance matrix $\hat{X}$, for a set of systems and a set of queries. Each pair indicates the estimated performance and associated uncertainty.

such that

$$\begin{aligned} var(M) = &\quad n^{-2} e^T \Sigma e \\ var(\hat{M}) = &\quad \beta^T (\Sigma + U) \beta \\ cov(M, M_\Phi) = &\quad n^{-1} \beta^T \Sigma e \end{aligned}$$

where $e = \{1\}^{n \times 1}$ is the vector of n components, each equal to 1, and $\Sigma = cov(X)$ is a $n \times n$ covariance matrix. The $(i,j)^{th}$ element of Σ is the covariance between the i^{th} and j^{th} columns of matrix X. In addition, $U = diag\big(E(v_1), ..., E(v_n)\big)$ is a diagonal matrix, referred to as the uncertainty matrix, also $E(v_q) = l^{-1} \sum_{i=1}^{l} v_{i,q}$ is the average uncertainty for query q.

Substituting for the variances and covariance, we have

$$\rho_\beta = \frac{\beta^T \Sigma e}{\{(e^T \Sigma e)(\beta^T (\Sigma + U)\beta)\}^{\frac{1}{2}}} \tag{7.2}$$

In addition, reordering Equation 7.2 gives

$$\gamma_\beta \equiv (e^T \Sigma e)^{\frac{1}{2}} \rho_\beta = \frac{e^T \Sigma \beta}{\{\beta^T (\Sigma + U)\beta\}^{\frac{1}{2}}} \tag{7.3}$$

Maximizing ρ_β is equivalent to maximizing γ_β since $(e^T \Sigma e)^{1/2}$ is a constant. The maximum value of Equation 7.3 can be approximated by the minimization problem that is expressed in a quadratic programming form [2] [MoWMMRCC67]:

$$\min_\beta \frac{1}{2} \beta^T (\Sigma + U)\beta - e^T \Sigma \beta \tag{7.4}$$

where the elements of Σ are approximated based on the available relevance judgments. In addition, the impact of uncertainty in approximating the covariance matrix Σ is captured by U matrix.

In the following section, we add the generalization constraint to the optimization in Equation 7.4 that enhances effective evaluation of new, previously unseen systems. We note that other constraints could easily be incorporated into this framework.

[2] The optimization form in Equation 7.3 is in convex-fractional form and is optimized by transferring it to quadratic programming form

7.3.1 Generalization Constraint

If all the relevant documents for each query in the test collection are identified, then the test collection generalizes to any system. Unfortunately, we can guarantee to identify all relevant documents only if we judge all the documents in the collection, which is prohibitively costly. Pooling documents significantly reduces the number of documents we need to judge, as discussed earlier. However, pooling does not guarantee that all the relevant documents have been identified. Clearly, the fewer unidentified relevant documents in the test collection, the more generalizable the test collection is. Thus, we define an optimization function that not only minimizes the difference between $\hat{M}$ and M vectors, but also minimizes the number of un-judged relevant documents.

We define $r \in R^{n\times 1}$ to hold the expected number of un-judged relevant documents for each of the n queries. Thus, r_j denotes the expected number of un-judged relevant documents for query q_j. At iteration k we allocate a part of the budget B_k to the j^{th} query that is proportional to β_j. Also, the number of newly judged relevant documents will be proportional to $\beta_j \times r_j$. The total number of relevant documents judged in the k^{th} stage is proportional to $\beta^T r$, ignoring the constant of proportionality. The linear function $\beta^T r$ is treated as a generalization constraint in our optimization. Clearly, we want to maximize the total number of relevant documents in order to achieve maximum generalizability. Using a Lagrange multiplier, $\lambda \geq 0$, we combine the constraint and the optimization function, defined in Equation 7.4, to obtain

$$\min_{\beta} \quad [\frac{1}{2}\beta^T(\Sigma + U)\beta - e^T\Sigma\beta - \lambda\beta^T r] \tag{7.5}$$

The optimization in Equation 7.5 is convex and can be solved by using a sequential quadratic programming algorithm [Mur88]. Section 7.4.4 discusses how to estimate the expected number of relevant documents r_j in practice.

7.4 Implementation Details

Before describing the experiments, we discuss a number of implementation issues. Note, however, that the setting of λ is discussed in Section 7.5.

7.4.1 Prioritizing Query-Document Pairs

The set of β coefficients that minimize the Equation 7.4 are considered as query priority scores for the next round of relevance judgments. In addition, a document selection method can be used to prioritize a set of documents that are returned in response to a query. In the simplest case, the prioritization of documents is determined by the pool depth and is adjusted according to the available budget. Thus, the document priority score is 1 if a document is in the pool and 0 otherwise.

However, several document selection techniques, e.g. [CAS06, AP08, YA06] have been recently proposed that enable more efficient prioritization of documents. For instance, Javed Aslam et al. [AP08] define a sampling distribution over documents based on their rank in a result list. Documents with higher rank are given higher probability to be selected. To prioritize documents, we use the same method as it is reported to be among the best available document selection approaches. Hence, a subset of docu-

ments are selected based on the sampling strategy and their priority scores are set to the corresponding probabilities. Also, the priority scores of non-selected queries are set to 0.

Thus, we define query-document priority scores as $w_{qd} = w_q \times w_d$ where w_q refers to the q^{th} element in β vector and w_d is the document priority score that is calculated by a document selection algorithm. Once the query-document priority scores are calculated, we select a subset of them with the highest priority scores that fit the available budget.

7.4.2 Estimating Covariance Matrix

At the k^{th} stage of the iterative process, the relevance judgments collected so far are used with the performance estimator proposed by Javed Aslam et al. [AP08] to approximate systems' effectiveness and form $\hat{X}$. The performance matrix $\hat{X}$ is then used to compute the covariance matrix Σ. If the set of l participating systems is known as a random sample of systems' space, we use the formulation explained in Section 4.5.1 to compute Σ. Alternatively, if there are some prior information about similarity between the participating systems and unseen systems, the formulation explained in Section 4.5.2 is used to compute Σ.

7.4.3 Estimating Uncertainty Matrix

The performance estimator proposed by Javed Aslam et al. [AP08] provides the variance of estimation that is due to unjudged documents in a rank list. We use the same variance measure to compute v_{sq} scores and form the uncertainty matrix U.

7.4.4 Estimating Unseen Relevant Documents

It is difficult to determine whether or not all relevant documents for a query have been judged. However, the prior work of Zobel [Zob98] suggests that some degree of estimation is possible, given an initial set of relevance judgments. He fitted the set of the relevance scores of the initial judgments with a power law distribution. Experimental results in [Zob98] demonstrated high prediction accuracy when estimating the total number of unseen relevant documents retrieved for all queries in a test collection.However, when predicting relevant documents for a single query, there was a large uncertainty in the estimates.

Alternatively, given a set of initially judged documents as a training set, we use a support vector machine (SVM) classifier [CV95] to partition unjudged documents into relevant and non-relevant categories. We use SVM because it is reported among the best performing classifiers in information retrieval experiments [BCYS07].

In order to train the classifier, we first extract features from each of the judged documents. The features are a set of relevance scores provided by the l participating systems. If a document is not retrieved by a participating systems, the associated relevance score is set to the minimum relevance score provided by that system for a retrieved document.

7.5 Evaluation Settings

Evaluations are conducted by comparing the performance of a set of IR systems based on the full set of queries and the full set of relevance judgments, with the systems' performance based on a limited

number of relevance judgments supplied by our budget allocation method.

We focus on incrementally building relevance judgments for the commonly pooled set of documents. The budgets for the initial and the refinement phases are allocated during the construction of the test collection and only documents are considered for constructing relevance judgments that have been retrieved by the participating systems.

In the evaluation of our approach, we consider both the accuracy of evaluating the performance of participating systems with additional relevance judgments and generalization to unseen systems. When evaluating the generalization of the budget allocation method, we define the criteria for identifying markedly different systems. We use the mean average reuse (MAR) [CGJM10] to characterize individual systems and select those with low MAR as new, yet unseen systems.

7.5.1 Baseline Methods

We consider three baseline methods for resource allocation in comparison with our resource optimization method which is referred to as Query-Document Prioritization (QDP):

- *Uniform Allocation* (UN), in which the available budget is uniformly allocated across queries. For example, if the budget can cover only 200 new judgments and there are 100 queries, we judge two new documents per query.

- *Random Allocation* (RA), in which a random set of n queries is selected and the budget B_2 is uniformly allocated across the selected queries. In our experiments we use n that corresponds to the number of queries selected by our optimization method. We repeat the random query sampling for 1000 trials and report the average of the corresponding results.

- *Subset Allocation* (SA), in which a subset of queries is selected based on the budget-constrained convex optimization introduced in Section 5.3. Similar to the QDP method, the subset allocation method uses an iterative process to collect relevance judgments. However, at each iteration, it selects a subset of queries based on the convex optimization in Equation 5.2, and then equally allocates the available budget across the selected queries.

7.5.2 Data Sets and Parameter Settings

Our experimental investigations were performed using two test collections: (*i*) the TREC 2004 Robust track consisting of 249 queries, 14 sites with a total of 110 automatic runs, and 311,410 relevance judgments, and (*ii*) the TREC-8 Ad-Hoc test collection consisting of 50 queries, 39 sites with 13 manual runs and 116 automatic runs, and 86,830 relevance judgments. Both test collections use TREC Disks 4 & 5, excluding the Congressional Record sub-collection.

For our purposes we consider each run as an individual IR system but take special care when considering IR systems from the same site. In particular, when experiments require that we exclude some of the systems in order to treat them as new, yet unseen systems, we hold out not only individual runs but the entire set of runs from the same site. Furthermore, during the computation of performance metrics, we remove documents that are uniquely retrieved by the held-out systems when that is required.

Automatic runs use automatic query formulation, while manual runs allow human to formulate queries. The latter runs typically perform better. Since all the runs in Robust track are automatic runs, we treat them as a homogeneous set of systems. Also, we treat the runs in TREC-8 as a heterogeneous set of systems because of the existence of both automatic and manual runs.

7.5.3 Experimental Setup

In order to test the generalization and robustness of a budget allocation method to evaluate new systems, we first divide the TREC runs into participating systems and new, still unseen systems that contribute new search results. To collect relevance judgments, we randomly select a few sites and use their corresponding runs as participating IR systems. Using the document selection technique proposed by Javed Aslam et al. [AP08] we select and evaluate the set of documents pooled by these participating systems.[3] The number of selected documents is adjusted to fit the budget allocated to the initialization step.

We split the held-out systems into two groups. For each held-out system, and each query, we compute the average reuse (AR) [CGJM10]. This measures the overlap between the documents retrieved by a held-out system and the judged documents. We then define the mean average reuse (MAR) for a held-out system as the average of AR values over the full set of queries.

Based on the MAR values, we split the held-out systems into two groups. The first group consists of systems with high MAR across runs. These systems can be evaluated using the existing relevance judgments. The second group, referred to as the new set, consists of runs that have low MAR. These systems require additional relevance judgments in order to be evaluated.

The budget B is divided in $p \geq 2$ portions $\{B_1, ..., B_p\}$. The portions of budget are spent to collect relevance judgments through an iterative process. The full experiment comprises the following steps:

- Initialization Phase:

  1. Pick s_1 percent of sites at random, these are the held-in sites.
  2. For each query, select a subset of documents retrieved by the held-in runs using the document selection approach and collect the associated relevance judgments. Compute the performance matrix $\hat{X}$. The number of selected documents is determined based on the budget allocated to the initialization stage, B_1. The budget is uniformly distributed across queries.
  3. Compute the MAR for the held-out runs. Average the MAR scores across runs from the same site and produce average reuse score for each site.
  4. Pick s_2 percent of sites with low MAR scores and treat their runs as new systems. The remaining runs are evaluated with the existing relevance judgments and their performance values are added to the matrix $\hat{X}$. Note, however, that the remaining runs do not contribute to the document pool.

- Expansion Phase: for k between 2 and p repeat the following steps.

[3] We used simple random sampling without replacement (SRSWO) as the sampling method to select a subset of documents. Also, the Horvitz-Thompson-type estimators [DGH52] was used to provide approximately unbiased estimates of x_{sq} performance scores.

– given the budget B_k, acquire additional relevance judgments for a subset of documents pooled by participating systems in one of the four ways:

1. *Uniform* (UN): for each of the n queries, acquire relevance judgments for an additional k_1 documents, where k_1 is adjusted based on B_k.
2. *Subset Allocation* (SA): select a subset of m queries by using the budget-constrained query selection optimization in Equation 5.2 and the available budget B_k, and acquire relevance judgments for additional k_2 documents per query, where $m \times k_2 = n \times k_1$.
3. *Random Allocation* (RA): for a random sample of m queries acquire relevance judgments for additional k_2 documents per query, where $m \times k_2 = n \times k_1$.
4. *Query-Document Optimization* (QDP): prioritize query-document pairs based on the method explained in Section 7.3. Order the query-document pairs and acquire relevance judgments for a subset of them that fit the budget B_k.

7.5.4 Lagrange Multiplier

The QDP formulation of the budget optimization in Equation 7.5 requires the computation of the Lagrange multiplier $\lambda \geq 0$. We determine λ empirically by systematic exploration of the range of values for $0 \leq \lambda \leq 10$. This is iteratively performed when expanding relevance judgments in stages between 2 and p.

During stage $k \geq 2$, we have allocated budgets $B_{(1:k)} = \sum_{i=1}^{k} B_i$ and acquired relevance judgments across queries. We then simulate the steps of the experiment listed in the previous section, where we split the budget $B_{(1:k)}$ into two parts $B^{(1)}_{(1:k)}$ and $B^{(2)}_{(1:k)}$ in the same proportion as true budget allocation $B_{(1:k)}$ and B_{k+1}.[4] During this simulation the estimated number of un-judged relevant documents,r_j, for a query q_j is set to the number of relevant documents identified during the stages between 1 and k, using the budget $B_{(1:k)}$ for query q_j. This ensures that at the expansion phase of the simulation to determine λ, no selected query requires more assessments than we have acquired so far. Thus, we have all the relevance judgments needed to evaluate the performance of the simulation.

For a particular value of λ within the range $0 \leq \lambda \leq 1$ we apply a 10-fold cross-validation technique [Koh95b]. In each of the 10 iterations, 10% of participating systems are held out (these become our simulated new systems). Relevant documents that are in the initial document pool but solely retrieved by the held-out systems are removed from the pool. The QDP method, using the reduced set of judgements, produces a set of query-document pairs. In our experiments, we separately optimize the value of λ for evaluating (*i*) participating systems and (*ii*) new systems. Thus, when evaluating participating systems, we assess λ by computing the Kendall-τ of the held-in systems' ranking with the corresponding ranking induced by using all the relevance judgments acquired using budget $B_{(1:k)}$. Also, when evaluating new systems, the Kendall-τ of ranking held-out systems is computed. For each experiment, we record the average Kendall-τ for the 10 trials. Finally, we choose the λ value with the highest average Kendall-τ.

[4]The budget $B_{(1:k)}$ is split between $B^{(1)}_{(1:k)}$ and $B^{(2)}_{(1:k)}$ such that $\frac{B_{(1:k)}}{B_{k+1}} = \frac{B^{(1)}_{(1:k)}}{B^{(2)}_{(1:k)}}$

Table 7.1: Result for Robust TREC 2004 runs evaluated by MAP. The first two columns report experimental parameters. The next columns report the Kendall-τ of (*i*) participating systems, and (*ii*) previously unseen systems for each resource allocation.

#	$(s_1, s_2)\%$	$(B_1, B_{(2:3)}) \times 10^3$	Kendall-τ							
			participating systems				new systems			
			UN	RA	SA	QDP	UN	RA	SA	QDP
1	(10,50)	(2,8)	0.63	0.58	0.65	**0.68**	0.54	0.51	0.59	0.58
2		(5,5)		0.61	0.70	**0.78**		0.52	0.66	**0.71**
3		(8,2)		0.63	0.67	**0.79**		0.52	0.63	**0.74**
4	(10,40)	(4,16)	0.72	0.66	0.76	**0.90**	0.68	0.62	0.70	**0.76**
5		(10,10)		0.68	0.79	**0.89**		0.65	0.77	**0.81**
6		(16,4)		0.74	0.81	**0.91**		0.67	0.74	**0.83**
7	(20,40)	(4,16)	0.79	0.69	0.83	**0.91**	0.80	0.66	0.74	**0.84**
8		(10,10)		0.75	0.82	**0.89**		0.67	0.80	**0.90**
9		(16,4)		0.77	0.83	**0.91**		0.70	0.81	**0.91**

7.6 Experimental Results

Our experimental results are divided into two parts, following the separate treatment of the homogenous and heterogeneous sets of IR systems. Thus, in Section 7.6.1 the homogeneous collection of Robust TREC is considered and the unbiased estimator explained in Section 4.5.1 are used to approximate Σ. For the Robust TREC test collection we report experiments using a total budget that covers either 10,000 or 20,000 relevance judgments.

In Section 7.6.2 we present experiments with the heterogeneous collection of TREC-8 and use manual runs as new systems. For the TREC-8 test collection we report results using a total budget that covers either 2,000 or 4,000 relevance judgments. This is less than 5% of the budget that covers 86,830 relevance judgments for the collection. In the implementation of QDP we use the unbiased estimator to approximate Σ for a weighted sample of systems introduced in Section 4.5.2.

7.6.1 Homogeneous Systems

We applied the steps explained in Section 7.5.3 across 10 trials and, in each trial we randomly chose $s_1\%$ of sites and associated runs as participating systems. We set $s_1 = 10\%$ or 20% that was equivalent to select between 15 to 30 runs as participating systems which was sufficiently large for the propose of our experiment. The remaining runs were evaluated for MAR and the $s_2\%$ of sites with the lowest MAR scores were chosen to be new systems. Depending on the average MAR scores, s_2 varies between 50% and 40% of the total number of sites. We reported averages over the 10 trials.

We repeated the experiment for 3 different values of s_1 and s_2, and 3 different budget allocations. The available budget was first divided in two portions. The first portion was used as B_1 at the initialization step. The second portion, denoted as $B_{(2:3)}$, was equally divided in two parts. Each part was used in an iteration of the expansion phase.[5] Table 7.1 summarizes the results.

We report the Kendall-τ statistic between the ranking of the systems induced by a resource allocation method, and the ranking scores of the systems over the full set of queries and the full set of relevance

[5]We also repeated the experiments with dividing the second part of the budget into smaller portions to increase the number of iterations of the expansion phase. However, no improvement was obtained over the case with only two iterations. It is also worth to try some complex allocation, e.g. using an exponential distribution across the iteration. This will remain as future work.

judgments in the original test collection. We report separate Kendall-τ statistics for participating systems and for new systems, which is common in the literature and permits us to separately discuss the accuracy and generalization of the methods.

We observe that for all 9 experimental configurations, the Kendall-τ scores of the QDP method outperform the other three budget allocation methods. The uniform allocation strategy is comparable and often better than the random allocation strategy for both participating and new systems. The subset allocation (SA) method outperforms the uniform allocation when $s_1 = 10\%$ (rows 1 through 6). However, for $s_1 = 20\%$ the SA method performs no better than a uniform allocation for new systems, but remains better for participating systems. In contrast, the QDP method is superior in all cases except for configuration 1 in which the initial budget B_1 is only 2000 relevance judgments. We believe this is due to the small value of B_1 which only covers 0.6% of the total assessor judgments.

It is important to note that the QDP method has significantly better Kendall-τ scores than the random allocation method, for both participating and new systems, indication that the optimization achieved both accuracy and generalizability.

Increasing the number of participating systems s_1 with the same budgets B_1 and $B_{(2:3)}$ leads to a larger improvement in Kendall-τ of new systems' ranking than increasing the budgets and keeping the number of participating systems s_1 constant. This can be seen by comparing experimental configurations 5 & 8 or 6 & 9. As a result, a higher diversity of participating systems results in a better ranking of new systems.

When prioritizing queries by the QDP method, we separately optimized λ for evaluating participating systems and new systems as discussed in Section 7.5.4. Figure 7.3 shows the optimal value of λ, computed after the initialization step and before spending B_2, across the various configurations of B_1. In all the configurations, the optimal λ for ranking participating systems was smaller than the optimal λ obtained for ranking new systems. This is intuitively clear, as larger values of λ let the generalization constraint contributes more in the optimization process and causes a better ranking of new systems.

Before running the experiments, we anticipated that the optimal λ for participating systems is 0, meaning no contribution of the generalization constraint to the optimization process. Thus, the optimization is concentrated on the first part that maximizes the accuracy of ranking participating systems. However, the optimal λ for participating systems was bigger than zero for various B_1. However, as B_1 increased the optimal λ decreased toward 0. This suggests that when the initial budget is very small, the generalization constraint effectively improves not only the ranking of new systems but also the ranking of participating systems.

In the experiments conducted in this section, the set of participating and new systems were randomly chosen. We therefore used unbiased estimator of matrix Σ, as explained in Section 4.5.1. In the next section, we consider the scenario in which participating and new systems are not randomly chosen. Rather, we consider a set of highly performing systems as new systems and use the appropriate unbiased estimators discussed in Section 4.5.2.

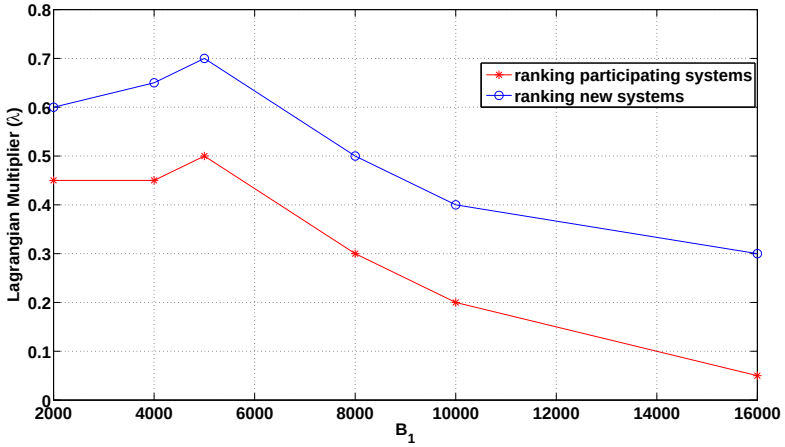


Figure 7.3: The optimum value of lagrangian multiplier, λ, obtained for various B_1. The optimum λ is adjusted as discussed in Section 7.5.4.

7.6.2 Heterogeneous Systems

The TREC-8 test collection consists of 129 runs of which 13 runs are manually tuned and outperform the automatic runs. The 11 best performing runs are all manual and their performance measured by MAP is statistically significantly better than the remaining runs. We consider the 13 manual runs as new (unseen) systems and the rest as participating systems. We consider two variants of the QDP method. The first variant is as same as the QDP method used in previous section for which the unbiased estimator of a random sample of systems, as introduced in Section 4.5.1, is used to compute the covariance matrix Σ. The second method, denoted as QDP*, uses the unbiased estimator of a weighted sample of systems, as explained in Section 4.5.2, to compute Σ. Thus, in contrast to QDP, the participating systems contribute non-uniformly in prioritizing queries in QDP*. The intuition is that, since the new systems are likely to perform better than participating systems, we may achieve better generalization, if we preferentially weigh highly performing participating systems.

At the initialization stage of QDP* method, all the automatic systems equally contribute to select a subset of documents for constructing associated relevance judgments. Next, k participating systems with the highest average performance scores are selected. If the i^{th} system is among the selected ones, the corresponding weight is $p_i = \frac{1}{k}$, otherwise $p_i = 0$. Further, only documents retrieved by the systems with $p > 0$ are considered for query-document prioritization at the expansion phase.

We set $k = 30$ since (i) it was sufficiently large to approximate Σ matrix, and (ii) considering only 30 top performing out of 116 runs ensures that the sample of participating systems have relatively good performance. We repeated the experiment for 6 different budget configurations. The results are shown in Table 7.2.

For all budget configurations, the QDP method obtains the best Kendall-τ for ranking participating systems. However, QDP* outperforms the QDP and the other budget allocation method for ranking the new systems. Also, the SA method outperforms the UN and RA methods for the first three configurations. However, for budget configurations 4-6, UN method outperforms both RA and SA.

Table 7.2: Accuracy and Generalization of ranking systems in TREC-8 by Kendall-τ correlation. The 13 manual runs are treated as new (unseen) systems and 116 automatic runs are treated as participating systems. The QDP* is the extension of QDP method in which the unbiased estimators of a weighted sampling of systems are used to approximate covariance matrix Σ.

#	$(B_1, B_{(2:3)}) \times 10^3$	Kendall-τ									
		participating systems					new systems				
		UN	RA	SA	QDP	QDP*	UN	RA	SA	QDP	QDP*
1	$(\frac{1}{2}, \frac{3}{2})$		0.55	0.71	**80**	0.78		0.20	0.30	0.32	**0.54**
2	(1,1)	0.61	0.57	0.75	**0.81**	0.80	0.34	0.27	0.44	0.39	**0.63**
3	$(\frac{3}{2}, \frac{1}{2})$		0.6	0.76	**0.83**	0.82		0.28	0.39	0.39	**0.67**
4	(1,3)		0.65	0.84	**0.92**	0.90		0.48	0.47	0.50	**0.78**
5	(2,2)	0.86	0.69	0.83	**0.91**	0.89	0.69	0.49	0.62	0.68	**0.87**
6	(3,1)		0.75	0.84	**0.92**	0.90		0.51	0.66	0.69	**0.91**

Table 7.3: Root Mean Squared Error (RMSE) results for TREC-8 test collection. The 13 manual runs are treated as new (unseen) systems and 116 automatic runs are treated as participating systems. QDP* is the extension of QDP in which the unbiased estimators of a weighted sampling of systems are used to approximate covariance matrix Σ.

#	$(B_1, B_{(2:3)}) \times 10^3$	RMSE									
		participating systems					new systems				
		UN	RA	SA	QDP	QDP*	UN	RA	SA	QDP	QDP*
1	$(\frac{1}{2}, \frac{3}{2})$		0.22	0.14	**0.09**	0.1		0.48	0.45	0.47	**0.3**
2	(1,1)	0.17	0.19	0.11	**0.07**	0.09	0.46	0.46	0.39	0.40	**0.27**
3	$(\frac{3}{2}, \frac{1}{2})$		0.18	0.12	**0.07**	0.08		0.42	0.41	0.40	**0.24**
4	(1,3)		0.17	0.10	**0.06**	0.08		0.38	0.30	0.27	**0.20**
5	(2,2)	0.14	0.14	0.09	**0.05**	0.09	0.34	0.35	0.26	0.25	**0.16**
6	(3,1)		0.16	0.08	0.06	0.06		0.36	0.27	0.24	**0.16**

We also report root mean squared error (RMSE) between the MAP scores of the systems estimated based on a budget allocation method and the true MAP scores measured over the full set of relevance judgments in TREC-8. Once again, separate scores are provided for participating and new systems. As shown in Table 7.3 similar observations hold true for RMSE.

However, the RMSE scores obtained for new systems are considerably larger than the RMSE scores obtained for the participating systems. This is because the manual runs retrieve 24% of the unique relevant documents that were judged in the original document pools. This means many relevant documents, retrieved by the manual runs, are absent from the document pools. Thus, it is impossible to accurately measure the absolute performance of the manual systems even after judging all the documents pooled by the participating systems.

Clearly, if new systems are retrieving a substantial number of unique relevant documents, we cannot expect to approximate their absolute performance well, unless we can afford to acquire additional relevance judgments for previously unseen documents that are retrieved by the new systems. This scenario was already considered in Chapter 5.

7.7 Summary

We considered the problem of prioritizing query-document pairs for relevance assessments given a budget constraint, in order to (*i*) improve the accuracy of evaluating participating systems, and (*ii*) ensure

that the test collection generalizes to new, previously unseen systems. We proposed an iterative procedure for collecting relevance judgments. In the initialization phase, we allocated a budget B_1 uniformly across all queries, acquiring a corresponding set of relevance judgments. In the expansion phase, we iteratively used an optimization framework to prioritize query-document pairs and optimally allocate the remaining budget.

The novelty of the method was in (*i*) combing the query selection and document selection approaches to form a unified budget allocation through explicit cost optimization, and (*ii*) formulating the problem as a convex optimization for which computationally efficient algorithms exist. Our experiments compared the QDP method with, uniform, random sampling and subset allocation methods. They provided strong evidence that the QDP method is superior to the selected baseline methods in (*i*) measuring the performance of participating systems, and (*ii*) generalizing to new, as yet unseen systems.

Chapter 8

Crowdsourcing Relevance Judgments

We consider the problem of acquiring relevance judgements for information retrieval test collections through crowdsourcing experiments. We collect multiple, possibly noisy relevance labels per document from workers of unknown labeling accuracy. We use these labels to infer the document relevance based on two methods. The first method is the commonly used majority voting (MV) which determines the document relevance based on the label that received the most votes, treating all the workers equally. The second is a probabilistic model that concurrently estimates the document relevance and the workers accuracy using the expectation maximization (EM). We run simulations and conduct experiments with crowdsourced relevance labels from the INEX 2010 Book Search track to investigate the accuracy and robustness of the relevance assessments to the noisy labels. We also observe the effect of the derived relevance judgments on the ranking of the search systems. Our experimental results show that the EM method outperforms the MV method in the accuracy of relevance assessments and IR systems ranking. The performance improvements are especially noticeable when the number of labels per document is small and the labels are of varied quality.

8.1 Introduction

Relevance judgments are manually constructed by a set of human assessors. Traditionally, the assessors are trained experts. However, as the corpus and the number of queries grow, the cost of acquiring relevance judgments from expert assessors for a sufficiently large number of documents becomes prohibitive. In response to this problem, the IR community has recently been exploring the use of crowdsourcing services to obtain relevance judgments at scale.

Web services, such as Amazon Mechanical Turk [1], facilitate the collection of relevance judgments by temporarily hiring thousands of crowd workers. While the labels provided by the workers are relatively inexpensive to acquire, they vary in quality, introducing noise into the relevance judgments and, consequently, causing inaccuracies in the system evaluation [KKKMF11]. In order to address the issue of noisy assessments, it is common to collect multiple labels per document from different workers, in the hope that the consensus across multiple labels would lead to more accurate relevance judgments.

We assume that a set of labels is collected for each document from multiple crowd workers and

[1] www.mturk.com

that the accuracy of each worker is unknown, as is the true relevance of the documents. A probabilistic model is suggested for estimating both the relevance of the documents and the workers accuracy. We implement the probabilistic model by using the expectation maximization algorithm (EM) as in [DS79]. The performance of the probabilistic model (EM) is particularly compared with the performance of the majority voting (MV) that has been frequently used for label aggregation in IR, e.g. [KKKMF11, AM09, SJ11b].

The experiments are conducted based on the crowdsourced labels from the INEX 2010 Book Search track to compare the MV and EM methods. We consider crowdsourced labels from two task designs that lead to different level of noise and observe their effect on estimating relevance judgments and systems ranking. Our empirical evidence shows that the EM method offers more reliable relevance estimations and systems ranking than the MV method, especially when labels collected for a document are few or varied in quality.

8.2 Assessment Errors

Relevance assessment errors in IR test collections have been considered by the IR community since the early Cranfield experiments [CK67]. Voorhees [Voo98] studied the effects of variability in relevance judgments on the stability of the comparative IR systems evaluation. She considered three sets of relevance judgments provided by three different sets of assessors for the TREC-4 test collection. She observed that there are about 30% disagreements between the labels provided by assessors of different groups. She also explored the effects of the judgments inconsistency on the ranking of the systems that participated in TREC-4 and observed no significant changes in the systems ranking. This was attributed to the stability of the average precision (AP) metric [BV00] that was used to evaluate the systems performance. Indeed, AP is calculated based on deep pools of documents obtained from the participating systems. Thus, some incorrect judgments in a ranked list do not significantly affect the values of AP and, therefore, do not perturb the ordering of the systems. In our experiments in Section 8.4, we confirm that a deep pool of judged documents can reduce the effect of noisy crowdsourced labels in the systems evaluation.

Recent trends in IR evaluations involve the use of large numbers of queries to enhance the reliability of the evaluation [CPK+08] while reducing the pool depths and, with that, the cost of acquiring relevance judgments [CAS06]. However, the use of recall-sensitive metrics, e.g. AP, with shallow document pools becomes more sensitive to assessment errors and leads to significant changes in systems rankings [CS10]. This has motivated studies of the factors that cause assessment errors such as the level of assessors expertise [BCS+08], the presentation of the documents for assessment, such as the sequence in which the documents are shown to the assessors e.g. [KKKMF11, STS11], and the assessors behavior [CS10].

Awareness of the assessment errors has further increased with the use of crowdsourcing services to supplement or replace the traditional ways of collecting relevance judgments. In crowdsourcing, the relevance assessment task is expressed in terms of a human intelligence task (HIT) that is presented to crowd workers through a crowdsourcing platform to solicit their engagement, typically for a specified fee. The effectiveness of the crowdsourcing approach has been investigated in terms of various factors,

including (*i*) the agreement with relevance judgments from trusted assessors [SJ11b], (*ii*) quality assurance techniques for detecting and removing unreliable workers [KKKMF11], and (*iii*) the cost incurred due to redundant relevance assessments that are needed for quality assurance, e.g. [MW10, SOJN08].

The use of multiple labels per document to improve the quality of relevance judgments involves label aggregation across the assessors, e.g. by arriving at a consensus through majority voting [AM09, SJ11a]. The effectiveness of the consensus approach has been assessed by Kazai et al. [KKKMF11] for IR tasks involving TREC and INEX test collections. Kumar and Lease [KL11] investigated the relationship between the document relevance and the workers accuracy by using a set of documents with known relevance as training data for a naïve Bayes method. The trained model estimated the relevance of new documents by aggregating labels based on worker accuracy.

8.3 Aggregating Multiple Labels

Consider a set of v documents and a set of w workers that provide relevance labels for the documents. We assume that the relevance of a document is a discrete variable with values in $\{0, 1, ..., G\}$. If the relevance value of i^{th} document is k ($k \in \{0, 1, ..., G\}$), then its $G+1$ dimensional vector R_i is a binary vector with k^{th} component 1 and the rest 0, i.e. $R_{ik} = 1$ and $R_{ij} = 0$, ($\forall j \neq k$). We now define a matrix $R \in \{0, 1, , G\}^{v \times (G+1)}$ of all the relevance vectors, comprising v binary R_i vectors.

Now consider a set of w workers with the corresponding accuracies $A = \{a_1, a_2, ..., a_w\}$, where a_j represents the accuracy of j^{th} worker. Both the document relevance R and the workers accuracy A are unknown to us. Instead, we have a set of relevance labels provided by the workers, i.e. $l_{ij} \in \{0, 1, ..., G\}$ is a relevance assessment of the document i by the worker j. A worker may provide relevance labels for some or all the documents. The goal is to estimate the true relevance value of the documents and the workers accuracy from a given set of labels. We assume that each document receives at least one label and the accuracy of the labels is unknown. Thus, in contrast to Kumar and Lease [KL11], we assume no initial information regarding the workers accuracy or the relevance of the documents.

8.3.1 Majority Voting

Consider a document i with the corresponding labels provided by a set of workers. Let n_{ig} be the number of times the document i is labeled as $g \in \{0, 1, ..., G\}$ by a set of workers. The majority voting assigns g as the document's true relevance label if n_{ig} is maximum.

8.3.2 Concurrent Estimation of Relevance and Accuracy

As an alternative to MV we consider the EM method for concurrent estimation of the document relevance and the workers accuracy. In this method the document relevance R and the workers accuracy A are unknown variables and the labels L provided by the workers are the observed data.

We take the same approach as [DS79] and consider the label aggregation model that assigns to each worker a $(G + 1) \times (G + 1)$ latent confusion matrix [Hub74] where $G + 1$ is the number of relevance grades. Each row refers to the true relevance value and each column refers to a relevance value assigned by a worker. Once the confusion matrix is calculated, we can determine the worker expertise based on metrics such as accuracy, the true positive ratio and the true negative ratio [SJ11a].

Let π^j_{gy} , $(\forall g \& l \in \{0, ..., G\})$ be the probability that the worker j provides a label y given that g is the true relevance value of an arbitrary document. The probability π^j_{gy} is computed based on the confusion matrix for the worker j. One estimator of π^j_{gy} is:

$$\pi^j_{gy} = \frac{\text{number of times worker } j \text{ provides label } y \text{ while the correct label is } g}{\text{number of labels provided by worker } j \text{ for documents of relevance } g} \tag{8.1}$$

where

$$\sum_{y=0}^{G} \pi^j_{gy} = 1 \qquad (\forall g \in \{0, ..., G\}), \text{ and } j \in \{1, ..., w\}$$

Of course, the calculation of π^j_{gy} assumes that R is known. In the following we show how π^j_{gy} and R can be simultaneously estimated.

Let p_g be the probability that a document drawn at random has a true relevance grade of g ($p_g = Pr[R_{ig} = 1]; i \in \{1, ..., v\}$). Now let n^j_{iy} be the number of times worker j provides label y for document i; for our purpose n^j_{iy} is binary, so if a worker labels the document $n^j_{iy} = 1$, otherwise $n^j_{iy} = 0$. If g is the true relevance grade of document i, $R_{ig} = 1$, then the probability of the worker j giving a grade y is π^j_{gy} and the probability of doing so n^j_{iy} times is $(\pi^j_{gy})^{(n^j_{iy})}$. Thus, the number of labels of each grade $\{0, 1, ..., G\}$ provided by worker j is distributed according to a multinomial distribution [EHP00] and its likelihood is proportional to

$$Pr(n^j_{i0}, ..., n^j_{iG}; \pi^j_{g0}, ..., \pi^j_{gG} | R_{ig} = 1) \propto \prod_{y=0}^{G} (\pi^j_{gy})^{n^j_{iy}} \tag{8.2}$$

Under the assumption that w workers independently label documents, the likelihood of labels provided for document i when $R_{ig} = 1$ is also proportional to

$$\prod_{j=1}^{w} Pr(n^j_{i0}, ..., n^j_{iG}; \pi^j_{g0}, ..., \pi^j_{gG} | R_{ig} = 1) \propto \prod_{j=1}^{w} \prod_{y=0}^{G} (\pi^j_{gy})^{n^j_{iy}}$$

Since the value of g is unknown, we compute the expectation of $Pr(n^j_{i0}, ..., n^j_{iG}; \pi^j_{g0}, ..., \pi^j_{gG} | R_{ig} = 1)$ over all possible values of g, i.e. we compute the marginal probability over all possible values of g:

$$\sum_{k=0}^{G} p_k \prod_{j=1}^{w} \prod_{y=0}^{G} (\pi^j_{ky})^{n^j_{iy}} \tag{8.3}$$

Also as the data from all documents are assumed to be independent, the joint probability distribution over all the v documents is

$$\prod_{i=1}^{v} \left(\sum_{k=0}^{G} p_k \prod_{j=1}^{w} \prod_{y=0}^{G} (\pi^j_{ky})^{n^j_{iy}} \right) \tag{8.4}$$

Equation 8.4 comprises mixtures of multinomial distributions. In order to estimate the quantities of interest, p_k, π^j_{ky} and R_{ig}, we apply expectation maximization (EM) [DS79]. In the EM algorithm we

treat π^j_{ky} and p_k as model parameters and R_{ik} as missing data. The EM algorithm then involves the following steps:

- Initialize R_{ik} values, e.g. randomly choose g and set $R_{ig} = 1$, and $R_{ik} = 0$ $(\forall k \neq g)$.
- Given the current estimate of R_{ik}, compute the maximum likelihood estimates of π^j_{ky} and p_k, as

$$\hat{\pi}^j_{ky} = \frac{\sum_{i=1}^{v} R_{ik} n^j_{iy}}{\sum_{y=0}^{G} \sum_{i=1}^{v} R_{ik} n^{iy}_j}; \qquad \hat{p}_k = \frac{\sum_{i=1}^{v} R_{ik}}{v}. \tag{8.5}$$

- Calculate the new estimate of $R_{ig} (\forall g \in \{1, ..., G\})$ based on $\hat{\pi}^j_{ky}$ and $\hat{p}_k$, as

$$Pr(R_{ig} = 1 | n^{\forall j}_{i0}, ..., n^{\forall j}_{iG}; \pi^{\forall j}_{g0}, ..., \pi^{\forall j}_{gG}) = \frac{p_g \prod_{j=1}^{w} \prod_{y=0}^{G} (\pi^j_{gy})^{n^j_{iy}}}{\sum_{k=0}^{G} p_k \prod_{j=1}^{w} \prod_{y=0}^{G} (\pi^j_{ky})^{n^j_{iy}}} \tag{8.6}$$

- Repeat steps 2 and 3 until the results converge.
- Finally, for each document i, set $R_{ig} = 1$ for the g with the maximum probability as calculated in equation 8.6, and $R_{ik} = 0 (\forall k \neq g)$.

Note that by combining π^j_{ky} values we can compute the accuracy of the worker j or other statistics of interest, e.g. the true positive ratio. Accuracy is estimated as $\hat{a_j} = \frac{\sum_{y=0}^{G} \hat{\pi}^j_{yy}}{\sum_{y,k} \hat{\pi}^j_{yk}}$.

8.4 Experiments

In this section we describe a set of experiments that compare the aggregation of relevance labels based on the MV and EM methods and the implications for the IR systems evaluation. The experiments are based on both synthetic and crowdsourcing data collected for INEX 2010 Book Search evaluation track.[2]

In the first experiment we use synthetic data and simulate the characteristics of the MV and EM methods. In the second experiment we assess the performance of the two methods based on crowdsourcing data. We then assess the accuracy of the MV and EM relevance assessments relative to the INEX official judgments. In the third experiment we investigate the impact of MV and EM relevance judgments on the system ranking using several performance metrics.

8.4.1 Experimental Data

In our experiments, we use the test collection and crowdsourced relevance data from the INEX 2010 Book Search evaluation track [KKKMF11]. The test collection comprises 50,239 books containing over 17 million scanned pages and 21 test queries (topics) with 169 judged pages per query, on average. This amounts to 3,557 judged pages that serve as a gold standard set for IR systems evaluation. Each page is assigned a relevance judgment based on four grades $\{0, 1, 2, 3\}$.

8.4.2 Crowdsourcing Experiments

Crowdsourced labels were collected for INEX 2010 Book Track Search task by using Mechanical Turk platform. For a given query, the user had to confirm whether the presented book page contains an answer

[2] http://www.inex.otago.ac.nz/tracks/books/books.asp

to the search query. A search query and corresponding pages were presented to the crowd workers for relevance judgments in the form of Human Intelligence Tasks (HITs). Each HIT consisted of 10 pages including up to 3 pages judged as relevant by the INEX assessors. Two HIT designs, referred to as 'simple' HIT and 'full' HIT design, were used to control the workers behavior and with that the label accuracy.

The simple HIT design included a minimal quality control using a single test question to capture random assignment of relevance labels by a worker. Furthermore, all the HITs were presented to a worker in a single batch, using the same generic HIT title, description, and keyword.

The full HIT design included several quality controls and qualified workers at different stages of the task. Since the HIT titles have an effect on the workers recruitment, the full HITs were grouped into 21 query-specific batches and included query details in the title, description, and keywords. This was likely to encourage workers who were interested in and knowledgeable about a particular query. Each HIT included two test questions to detect sloppy behavior: (*i*) 'please tick here if you did NOT read the instructions' at the top of the HIT form, and (*ii*) 'I did not pay attention' as a relevance label option. Furthermore, to enforce the requirement that the workers needed to read a page before deciding about its relevance, a captcha[3] was included asking them to enter the first word of the sentence that confirmed or refuted the relevance of the page.

On average, 6 labels from distinct workers were collected per document, 3 labels by the simple HIT and 3 labels by the full HIT. That amounts to 2179 labels for 727 query-document pairs from the simple HIT and 2060 labels for 683 query-document pairs from the full HIT. Also, 98% of query-document pairs labeled in the full HIT were among those labeled in the simple HIT. The workers were paid $0.25 to complete a simple HIT task and $0.50 for a full HIT task.

For evaluation of the relevance labels obtained by the MV and EM methods we consider three commonly used measures [SJ11b]: (*i*) the accuracy: the proportion of judged documents that are assigned the correct relevance label, (*ii*) the *true positive ratio* (TPR): the proportion of judged relevant documents that are correctly assigned the 'relevant' label, and (*iii*) the *true negative ratio* (TNR): the proportion of judged non-relevant documents that are correctly assigned the 'non-relevant' label.

8.4.3 Simulation

We conduct simulations of multiple labels aggregation to investigate the effects of (*i*) the number of labels collected for a document, and (*ii*) workers expertise on the performance of the MV and EM methods. We consider a set of 1,000 hypothetical documents with associated true relevance judgments. We also consider a set of 100 hypothetical workers, each with a particular level of expertise. We define workers expertise as their accuracy of labeling a randomly chosen document. Similarly to Carterette and Soboroff [CS10], we randomly sample the workers expertise from a Beta distribution and randomly assign documents to workers. We then apply the MV and EM methods to the collected labels in order to estimate the relevance of the documents. We use the measures defined in Section 8.4.1 to assess the performance of the two methods.

[3] http://www.captcha.net/

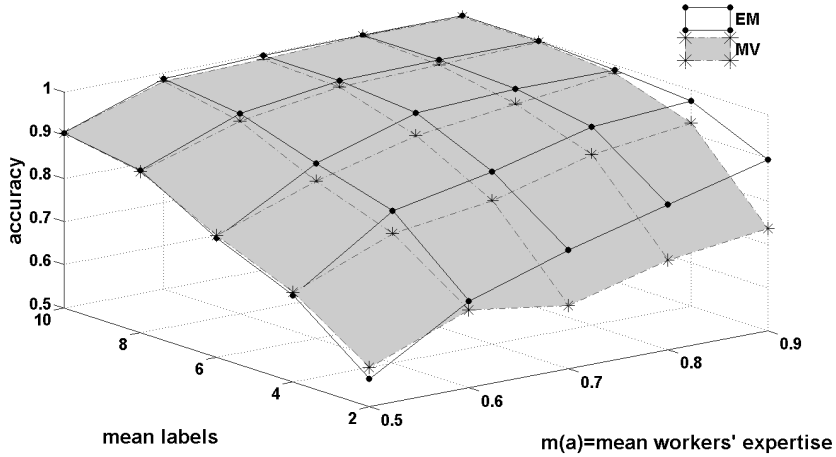


Figure 8.1: Comparisons of the accuracy of majority voting (MV) and expectation maximization (EM) for various numbers of labels collected per documents and different levels of assessors expertise (reliability).

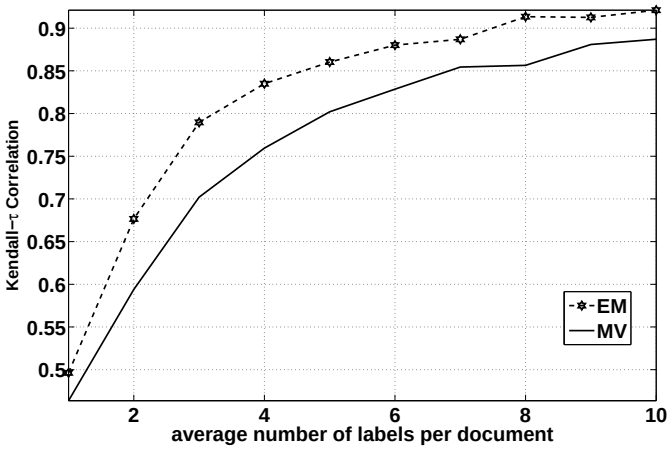


Figure 8.2: Kendall-τ correlation between the ranking of assessors true and estimated level of expertise.

We repeat the simulations by varying (*i*) the mean $m(a)$ of the Beta distribution from which a worker expertise is drawn, and (*ii*) the average number of labels collected per document. The results are shown in Figure 8.1 where the workers mean expertise varies between 0.5 and 0.9 and the average number of labels varies between 1 and 10.

When the workers average expertise is nearly random, $m(a)$=0.5, and the average number of labels per document is only 2, both methods exhibit poor accuracy. As the number of labels or the level of expertise increases, the performance of both methods improve. When the number of labels per document is small, e.g. 2 or 4 labels, but the workers average expertise is increased to 0.6 or higher, the EM method outperforms the MV. Finally, as the number of labels approaches 10 and the workers average expertise increases to 1, both methods obtain perfect accuracy. The simulations clearly show that the EM approach generally performs the same or better than MV.

We also assessed the performance of the MV and EM methods in estimating the workers accuracy. We use the gold standard set to determine the workers true accuracy and compare with the estimated

Table 8.1: Comparison of MV and EM relevance judgments based on (*i*) accuracy, (*ii*) true positive ratio (TPR) and (*iii*) true negative ratio (TNR). INEX 2010 relevance judgements are used as the gold standard set. Statistically significant differences are marked by †.

HIT	# Labels	accuracy		TPR		TNR	
		MV	EM	MV	EM	MV	EM
Simple	1000	0.57	**0.65**†	0.52	**0.67**†	0.60	**0.64**†
	2000	0.60	**0.68**†	0.54	**0.76**†	0.65	**0.68**†
	3000	0.67	**0.77**†	0.58	**0.79**†	0.70	**0.74**†
Full	1000	0.66	**0.69**†	0.72	**0.88**†	0.71	0.72
	2000	0.71	**0.78**†	0.78	**0.90**†	0.76	**0.78**†
	3000	0.80	**0.85**†	0.86	**0.93**†	0.84	0.82
Simple+Full	1000	0.66	**0.79**†	0.61	**0.85**†	0.62	**0.67**†
	2000	0.72	**0.80**†	0.66	**0.88**†	0.79	**0.74**†
	3000	0.76	**0.85**†	0.69	**0.91**†	0.75	**0.79**†

accuracies based on the relevance labels from the MV and EM methods. We compute Kendall-τ between the workers ranking induced by the EM or MV relevance judgments and the ranking based on the gold standard set. Figure 8.2 shows that for the set of workers with $m(a)$=0.7, EM outperforms MV for a range of labels per document. Similar results were also observed for $m(a)$=0.6, 0.8, and 0.9.

8.4.4 Relevance Agreement with INEX Judgments

We apply MV and EM to the labels collected from the two crowdsourcing experiments and compare the derived relevance judgments with the INEX gold standard set. We provide results for three sets of relevance judgments derived by: (*i*) the labels from the simple HIT, (*ii*) the labels from the full HIT, and (*iii*) the commentation of the both HIT. For each of the sets we use samples of 1000, 1500, or 2000 labels to estimate the document relevance. The samples are randomly selected but guaranteed that at least one label per document is included. To deal with sampling variance we report average performance of the methods over 10 random trials.

The experimental results are shown in Table 8.1 for each of the three evaluation measures, the accuracy, TPR, and TNR. In order to calculate TPR and TNR we assume that relevance judgments are binary and collapse labels $\{1, 2, 3\}$ into label 1. Statistically significant differences in the performance of the two methods are identified using a two-proportion z-test [SSR06] at the significance level of p=0.05.[4]

As seen in Table 8.1, for the labels from the simple HIT task, the EM method significantly outperforms MV across all the samples and evaluation measures. The average improvement of EM over MV is 0.06 in accuracy, 0.19 in TPR, and 0.04 in TNR. For the full HIT labels, the performance improvement of EM over MV is significant for most of the measures across the three samples. We do not get statistically significant difference only in two instances. The average performance improvement of EM across the three configurations is 0.04 in accuracy, 0.12 in TPR, and 0.003 in TNR. As seen, the average improvements obtained for the full HIT are relatively smaller than the average improvements obtained in the simple HIT. This is expected since the labels from the full HIT are of higher quality due to more elaborate quality assurances tests. Indeed, there is 70% agreement between the full HIT labels and the INEX official judgments compared to 55% for the labels from the simple HITs.

[4]Two-proportion z-test is a hypothesis test that determines whether the difference between two proportions is statistically significant.

Table 8.2: Kendall-τ scores for MV and EM rankings of 10 systems from the INEX 2010 Book Search track by using the precision at 5 different rank positions.

HIT	P@10		P@20		P@30		P@50		P@100	
	MV	EM	MV	EM	MV	EM	MV	EM	MV	EM
Simple	0.45	**0.62**	0.55	**0.71**	0.77	**0.89**	0.91	**0.98**	0.90	**0.99**
Full	0.68	**0.76**	0.71	**0.80**	0.88	*0.93*	1.00	1.00	0.99	0.99

Table 8.3: Kendall-τ scores for MV and EM rankings of 10 runs from the INEX 2010 Book Search track. The mean average precision (MAP) is calculated over all available judgments; stat-MAP is calculated for the subsets of documents using corresponding relevance judgments.

HIT	statMAP						MAP	
	10%		30%		50%			
	MV	EM	MV	EM	MV	EM	MV	EM
Simple	0.58	0.67	0.64	0.77	0.91	0.91	0.84	0.91
Full	0.66	0.72	0.67	0.79	0.80	0.89	0.79	0.87

This observation is consistent with the simulation results in Section 8.4.3. That is, when the labels are provided by quality workers and the number of labels is large, both MV and EM perform well. This can be seen for the accuracy scores of the full HIT in Table 8.1. When the number of labels is 1000 or 1500, the accuracy of EM is significantly higher than that of MV. However, for a larger sample of 2000 labels there is no significant difference between the accuracy scores.

Finally, we consider the combination of labels from the simple and the full HIT. For each sample size 50% of labels are randomly selected from the simple HIT labels and 50% from the full HIT labels. For all three samples and performance measures, the EM method shows statistically significant improvements over MV. The average improvement across the sample sizes is 0.09 in accuracy, 0.22 in TPR, and 0.05 in TNR, which are larger than the improvements for the simple HIT labels.

8.4.5 Impacts on Systems Ranking

We observe the effect of MV and EM relevance judgments on the system ranking. For the crowdsourced labels collected from the simple and the full HIT we apply MV and EM methods to create two sets of relevance judgments. Each set of relevance judgments are then used to measure the systems performance by an evaluation metric, e.g. average precision, and rank the average performance, e.g. mean average precision, of 10 retrieval systems that participated in the INEX 2010 prove it task. We compare the systems based on the precision metric at the rank position $\{10, 20, 30, 50 \text{ and } 100\}$.

Table 8.2 summarizes the Kendall-τ correlations between the ranking of systems based on the INEX official judgments and the ranking that is based on the relevance judgments inferred by MV or EM. For all the rank positions, the rank correlation is higher for EM than for MV. The average improvement for EM across the five rank positions is 0.12 for simple HIT and 0.04 for the full HIT labels.

Generally, we see a considerable effect of the cut-off level (rank position) on Kendall-τ. This is expected since when the cut-off level is small, e.g. p@10, even a few misjudged documents represents a high percentage of error and therefore significantly affects the ranking. As the cut-off level increases, for the same number of misjudged documents the percentage of error is relatively smaller and the ranking is not considerably affected.

We also explore the impacts of MV and EM on systems ranking when the average precision (AP) is used to evaluate the systems performance. The result is shown in Table 8.3. Once again EM outperforms MV for both the simple HIT and the full HIT labels. We investigate the effects of MV and EM on measuring AP with an incomplete set of relevance judgments, which is a common scenario in IR experiments [ACA+07]. We use the sampling technique used in [AP08] to select subsets of 10%, 30% or 50% of documents labeled by the crowd workers. We then apply MV and EM to the selected labels and use the statAP metric to estimate the AP scores. For each sample size we calculate statAP based on the corresponding INEX judgments, MV judgments, and EM judgments and obtain the system rankings. In Table 8.3 we show the rank correlations between the system ranking induced by the INEX official judgments and the system rankings induced by MV or EM. As seen, the EM method outperforms MV across the different sample size. As the sample size increases from 10% to 50%, the Kendall-τ scores increase correspondingly. The last column also represents the result of using the full set of labeled documents and the AP metric.

8.5 Summary

We considered the problem of creating relevance judgments using crowdsourcing experiments to collect multiple, possibly noisy, relevance labels for documents. We assumed that the workers' labels are varied in quality and of unknown accuracy. We also assumed that the true relevance judgments for documents are not available. We compared two methods for inferring document relevance from multiple noisy labels. The MV method treats all the workers equally and assigns the relevance label that has received the most votes. The EM method simultaneously infers document relevance and workers accuracy. We conducted a series of simulations with synthetic data and experiments with crowdsourced labels from the INEX 2010 Book Search track. Our experiments showed that the relevance judgments inferred by the EM method were the better estimations of true document relevance and lead to more accurate systems ranking. The EM performance improvements over MV were particularly noticeable when judgments were noisy and the number of relevance labels was small.

This work can be extended in several directions. In practice, some documents are easier than other documents to be labeled. Therefore, it will be interesting to take into account the document's difficulty when modeling a worker's accuracy. In the evaluation of systems performance we exploited the aggregation of noisy labels. However, the EM method provides estimation of the workers accuracy which can be used to grade workers and optimize the quality of additional labels by flittering the sloppy crowd workers from the pool of assessors. Furthermore, it can be used to compute workers pay based on the quality of their work. Finally, the full potential of the EM method could be realized through an iterative model of selecting workers and collecting relevance labels. Thus, it is beneficial to extend the crowdsourcing experiments and evaluate the dynamic and real time collection of relevance judgments.

Chapter 9

Conclusion

This dissertation explored three specific issues in order to construct cost-efficient test collections for information retrieval experiments. These are:

- Selection of a representative subset of queries to create relevance judgments and evaluate systems under budget constraints.
- Combination of the query selection and document selection approaches to efficiently create relevance judgments for a subset of query-document pairs.
- Integration of multiple noisy labels, collected by crowdsourcing experiments, to infer the relevance of a document.

This final chapter summarizes the results presented in earlier chapters of this dissertation, before considering possible future directions.

9.1 Results Summary

Chapter 4 assumed relevance judgments are available for all the queries in a test collection, and developed a theoretical framework for query selection. From the mathematical formulation it is implied that the optimal subset of queries should be least correlated with one another, but should have a strong correlation with the rest of queries. Finding the optimal subset of queries, even when relevance judgments are available and system's performance scores are known, is computationally intractable. Three query selection algorithms were discussed to implement the proposed query selection model in practice, namely: random sampling, the greedy algorithm, and convex optimization.

The quality of subsets selected by each of the three query selection algorithms were assessed in terms of (*i*) *accuracy* and (*ii*) *generalization*. Accuracy is concerned with how well a subset of queries can reproduce the relative performance of the participating systems when measured against the full set of queries. Generalization is concerned with how well the selected subset of queries can reliably evaluate a set of new systems, again compared to the full set of queries. The experiments were conducted using two TREC test collections, namely TREC-8 Ad-hoc track, and TREC 2004 Robust track.

We observed that both greedy and convex significantly outperformed the random sampling method in accuracy experiments. However, in generalization experiments, while the greedy method failed to

perform better than random sampling, the convex optimization consistently outperformed both greedy and random.

Chapter 5 showed that how the query selection approach can be used to enhance the reusability of a test collection. It was assumed that the initial set of relevance judgments is insufficient to reliably evaluate a set of new systems that did not contribute to pooling documents. A fixed budget was used to build some additional relevance judgments for the previously unjudged document retrieved by the new systems. The query selection approach was used to select a representative subset of queries, and then the budget was used to expand relevance judgments only for the selected queries. The experiment results on TREC 2004 Robust track showed that allocating the budget across a representative subset of queries leads to a better evaluation of new systems than uniformly allocating the budget across all the queries in a test collection.

Such a scenario should assist small groups of researcher investigating their new retrieval systems using large scale test collections, e.g. TREC Million Query track [ACA+07], where the initial set of relevance judgments is insufficient to reliably evaluate the new systems, and there is a limited budget to expand relevance judgments.

Chapter 6 relaxed the assumption that relevance judgments are available a priori, and extended the query selection framework to model uncertainty in the retrieval effectiveness metrics that are introduced by the absence of relevance judgments. Since the optimization was computationally intractable, an adaptive query selection algorithm was devised to provide an approximate solution. The effectiveness of the adaptive algorithm was demonstrated using various test collections, including a dataset of a commercial search engine with 1,000 queries and 30,000 relevance judgments. The experimental results showed that the adaptive method could reduce at least 35% of queries that were required by the considered baseline methods to obtain 90% accuracy in ranking the retrieval systems.

Chapter 7 extended the mathematical framework to combine the query selection and the document selection approaches, and devised a unified optimization framework. The unified optimization framework assigned a priority score to each candidate query-document pair and selected a subset of them to construct the associated relevance judgments under a budget constraint. The optimization framework assigned high priority scores to query-document pairs that could (*i*) accurately evaluate the relative performance of the participating systems, and (*ii*) generalize to new, previously unseen systems. We evaluated our optimization framework on two TREC test collections, namely TREC-8 Ad-hoc track and TREC 2004 Robust track. The experimental results showed that the optimization framework is cost efficient and yields a significant improvement in the generalization of the test collections.

Finally, Chapter 8 used crowdsourcing experiments to outsource the judgements task to a large number of assessors that are temporarily hired by using crowdsourcing, rather than assigning the task to a few well-trained experts. While the labels provided by crowdsourcing are relatively inexpensive, they vary in quality, introducing noise into the relevance judgments. To cope with noisy labels, it is common practice in information retrieval to collect multiple labels from different assessors and use majority voting to aggregate the labels. In contrast, we devised a probabilistic model that provided accurate relevance

judgments with a smaller number of labels collected per document. The effectiveness of the probabilistic method was assessed by using crowdsourced date collected for INEX 2010 book track.

9.2 Future Directions

While a large body of IR literature has studied the cost of test collections from various aspects, including this dissertation, there is still a lot to be done. We discuss the future directions in three lines:

- Formulating the mathematical framework with new objectives.
- Adding new optimization constraints to the optimization framework.
- Dynamic budget allocation in crowdsourcing experiments.

9.2.1 Objective Functions

Identifying characteristics of a representative subset of queries has surprisingly received little attention in IR literature. In this dissertation we focused on measuring the *relative* performance of systems and identified the properties of the representative subset by using Pearson linear correlation as the objective of our optimization framework (Chapter 4). Alternatively, we could investigate the properties of the representative subset of queries in terms of other objective functions.

For instance, the mean squared error (MSE) function could be used to assess systems in terms of their *absolute* performance. This is particularly important when our goal is to accurately measure a system's average performance rather than ranking a set of systems. Since the MSE function is quadratic, we can replace Pearson correlation with MSE as the objective for query selection, without violating the convexity requirements of the optimization framework. Thus, it is worth identifying the properties of the optimal subset of queries when the target of the query selection problem is defined as to minimize MSE.

9.2.2 Optimization Constraints

One of the main advantages of the convex optimization framework, used in Chapter 5 and Chapter 7, is its extensibility to accommodate various constraints. In Chapter 7 we defined a generalization constraint to reduce the probability that future as yet unseen systems return previously unjudged documents. Additionally, we can leverage research on identifying query characteristics that make queries better suited for use in systems evaluation and formulate new constraints within the optimization framework. By encoding such desirable constraints within our optimization framework, the method to identify a set of query-document pairs that embodies our requirements is a simple process. In the future, it is worth investigating a richer set of such heuristics, aiming to produce methods for test collection construction that are efficient, in terms of required resources for relevance assessments, and effective, in terms of accuracy of systems evaluations.

9.2.3 Dynamic Budget Allocation

The experimental set up can be expanded to examine the sensitivity of the optimization framework to errors in estimating the number of unjudged relevant documents as well as investigating the effects of

uncertainty that is due to (*i*) queries with no relevance judgements (Chapter 6), (*ii*) missing judgements (Chapter 5), or (*iii*) assessments' errors (Chapter 8).

Finally, the full potential of the method would be realized through an effective iterative model of relevance assessments in dynamic experiments. Thus, it is interesting to extend and evaluate the real-time applications of the cost optimization in the context of commercial search engines, e.g. Google and Bing, where queries and documents dynamically change over time.

Appendix A

List of Symbols

Capital letters (e.g. X and M) represent a matrix, or a vector. Lowercase letters (e.g. n) represent a scaler. Uppercase letters (e.g. $X^{l \times n}$) represent a dimension, and lowercase letters (e.g. μ_i) represent an index. Lowercase Greek letters (e.g. β) represent a parameter of a function. Uppercase Greek letters (e.g. Φ and Σ) represent a set or a matrix. However, if necessary, these rules may be violated.

S	The systems' population.
Q	The queries' population.
n	The number of queries.
l	The number of participating systems.
m	The number of selected queries.
Φ	The index set of selected queries.
Q_Φ	The subset of queries indexed in Φ.
$X \in R^{l \times n}$	A $l \times n$ system-query performance matrix.
$x_{i,j}$	The $(i,j)^{th}$ element of X matrix.
x_i	The i^{th} row of X matrix.
$M \in R^{l \times 1}$	The average performance column vector. The i^{th} element of M is the average of the corresponding row in X.
M_Φ	The average performance column vector calculated using the queries in Q_Φ.
$\alpha \in R^{1 \times n}$	The mean vector of X matrix. The j^{th} element of α is the mean of the column j in X.
Σ	The covariance matrix of X matrix.
$\sigma_{i,j}$	The $(i,j)^{th}$ element of Σ representing the covariance between columns i and j of X.
ρ_Φ	The Pearson linear correlation between M and M_ϕ vectors.
γ_Φ	The value of the optimization function that is maximized by selecting the optimal subset of queries.

μ	The average performance of a randomly chose system computed over the full set of queries.
μ_Φ	The average performance of a randomly chose system computed using Q_Φ query subset.
$e \in R^{n\times 1}$	A column vector of n ones.
$d \in \{0,1\}^{n\times 1}$	A binary column vector that indicates the selected queries. If query i is selected, $d_i = 1$, otherwise, $d_i = 0$.
$\beta \in R^{n\times 1}$	A column vector of n real values indicating the priority scores for the n queries.
$\| \, . \, \|_0$	The L_0 norm constraint that counts the number of non-zero elements in β or d and controls the size of the subset.
$\| \, . \, \|_1$	The L_1 norm that returns the sum of absolute values of the elements in β.
Ω	The cost of creating the complete set of relevance judgments.
B	The budget available for creating relevance judgments ($B \ll \Omega$).
B_i	The budget allocated to the stage i of an iterative budget allocation process.
$\hat{\alpha}$	An approximation of α computed based on a set of l participating systems.
$\hat{\Sigma}$	An approximation of Σ computed based on a set of l participating systems.
$\hat{X}$	An approximation of performance matrix X.
$\hat{x}_{i,j}$	An approximation of the performance score $x_{i,j}$.
$v_{i,j}$	The approximation variance of $x_{i,j}$.
$U \in R^{n\times n}$	A diagonal matrix, referred to as the uncertainty matrix. The $(i,i)^th$ element of U is the variances of systems' performance score calculated for query i.
f	The output of a classifier.
$\lambda \geq 0$	The Lagrangian multiplier that combines the quadratic component and a linear constraint of the optimization function.
w	The number of crowd assessors.
$R^{v\times(G+1)}$	A relevance matrix. In each row only one element is 1 indicating the relevance grade of the corresponding documents, and the rest are zero.
a_i	The accuracy of assessor i.
g	A relevance grade in $\{0, 1, ..., G\}$.

p_g	The probability that a document drawn at random has a true relevance grade of g.
π^j_{gy}	The probability that the worker j provides a label y given that g is the true relevance value of an arbitrary document.
n^j_{iy}	The number of times worker j provides label y for document i.

Appendix B

List of Acronyms

Leif Azzopardi: Include a list of Acronyms

AP	Average Precision
AR	Average Reusability
EM	Expectation Maximization
IQP	Iterative Query Prioritization
IR	Information Retrieval
MAP	Mean Average Precision
MAR	Mean Average Reusability
MV	Majority Voting
QDP	Query-Document Prioritization
QP	Query Prioritization
QS	Query Selection
RA	Random Allocation
SA	Score Adjustment

Appendix C

Mathematical Background

* Matrix

A matrix is a rectangular array of numbers, symbols or expression that is defined in terms of rows and columns. The individual items in a matrix are called elements.

* Column Vector

A column vector is a $n \times 1$ matrix, i.e. a matrix consisting of a single column.

* Row Vector

A row vector is a $1 \times n$ matrix, i.e. a matrix consisting of a single row.

* The Mean of a Vector

The mean of a vector is the average of its elements. The mean of X vector with n elements is:

$$\bar{X} = \frac{1}{n} \sum_{i=1}^{n} (X_i)$$

where X_i is the i^{th} element of X vector.

* The Variance of a Vector

The variance of a vector is a measure of how the elements of the vector are spread around its mean. The variance of X vector is calculated as:

$$\sigma_X^2 = \frac{1}{n} \sum_{i=1}^{n} (X_i - \bar{X})^2$$

If only m out of n elements of X are known, the sample variance is calculated as:

$$s_X^2 = \frac{1}{m-1} \sum_{i=1}^{m} (X_i - \bar{X})^2$$

* The Standard Deviation of a Vector

The standard deviation of a vector is the square root of its variance.

$$\sigma_X = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(X_i - \bar{X})^2}$$

* The Covariance of two Vectors

Covariance measures how much the elements of two vectors correspond with each other. The covariance between X and Y vector of the same dimension is

$$\sigma_{(X,Y)} = \frac{1}{n}\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})$$

If only m out of n elements of X are known, the sample covariance is calculated as:

$$s_{(X,Y)} = \frac{1}{m-1}\sum_{i=1}^{m}(X_i - \bar{X})(Y_i - \bar{Y})$$

* Mean Squared Error

The mean squared error is the mean of squared difference between the elements of two vectors of the same dimension. The mean squared error between X and Y is defined as:

$$MSE_{(X,Y)} = \frac{1}{n}\sum_{i=1}^{n}(X_i - Y_i)^2$$

* Pearson Linear Correlation

Pearson Linear correlation measure the dependence between the elements of two vectors. It is calculated by dividing the covariance of two vectors by the product of their standard deviation. The Pearson correlation between X and Y is defined as:

$$\rho_{(X,Y)} = \frac{cov(X,Y)}{\sqrt{var(X) \times var(Y)}}$$

If only m out of n elements of X and Y are known, the sample correlation coefficient is calculated as:

$$r_{(X,Y)} = \frac{\sum_{i=1}^{m}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{m}(X_i - \bar{X})^2 \sum_{i=1}^{m}(Y_i - \bar{Y})^2}}$$

* Kendall-τ Rank Correlation

Kendall-τ rank correlation measures the degree of correspondence between two rankings. Let R_Y and R_Y be the rank of elements in X and Y respectively. If X has n elements, there are $\frac{n(n-1)}{2}$ pairs of elements in total. Let n_c be the number of pairs that are in the same order in both R_X and R_Y. Also let n_d be the number of pairs that are in apposite order in R_X and R_Y. The Kendall-τ correlation between

R_X and R_Y is calculated as:

$$\tau_{(R_X,R_Y)} = \frac{2(n_c - n_d)}{n(n-1)}$$

Appendix D

Mathematical Background

* Matrix

A matrix is a rectangular array of numbers, symbols or expression that is defined in terms of rows and columns. The individual items in a matrix are called elements.

* Column Vector

A column vector is a $n \times 1$ matrix, i.e. a matrix consisting of a single column.

* Row Vector

A row vector is a $1 \times n$ matrix, i.e. a matrix consisting of a single row.

* The Mean of a Vector

The mean of a vector is the average of its elements. The mean of X vector with n elements is:

$$\bar{X} = \frac{1}{n} \sum_{i=1}^{n} (X_i)$$

where X_i is the i^{th} element of X vector.

* The Variance of a Vector

The variance of a vector is a measure of how the elements of the vector are spread around its mean. The variance of X vector is calculated as:

$$\sigma_X^2 = \frac{1}{n} \sum_{i=1}^{n} (X_i - \bar{X})^2$$

If only m out of n elements of X are known, the sample variance is calculated as:

$$s_X^2 = \frac{1}{m-1} \sum_{i=1}^{m} (X_i - \bar{X})^2$$

* The Standard Deviation of a Vector

The standard deviation of a vector is the square root of its variance.

$$\sigma_X = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(X_i - \bar{X})^2}$$

* The Covariance of two Vectors

Covariance measures how much the elements of two vectors correspond with each other. The covariance between X and Y vector of the same dimension is

$$\sigma_{(X,Y)} = \frac{1}{n}\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})$$

If only m out of n elements of X are known, the sample covariance is calculated as:

$$s_{(X,Y)} = \frac{1}{m-1}\sum_{i=1}^{m}(X_i - \bar{X})(Y_i - \bar{Y})$$

* Mean Squared Error

The mean squared error is the mean of squared difference between the elements of two vectors of the same dimension. The mean squared error between X and Y is defined as:

$$MSE_{(X,Y)} = \frac{1}{n}\sum_{i=1}^{n}(X_i - Y_i)^2$$

* Pearson Linear Correlation

Pearson Linear correlation measure the dependence between the elements of two vectors. It is calculated by dividing the covariance of two vectors by the product of their standard deviation. The Pearson correlation between X and Y is defined as:

$$\rho_{(X,Y)} = \frac{cov(X,Y)}{\sqrt{var(X) \times var(Y)}}$$

If only m out of n elements of X and Y are known, the sample correlation coefficient is calculated as:

$$r_{(X,Y)} = \frac{\sum_{i=1}^{m}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{m}(X_i - \bar{X})^2 \sum_{i=1}^{m}(Y_i - \bar{Y})^2}}$$

* Kendall-τ Rank Correlation

Kendall-τ rank correlation measures the degree of correspondence between two rankings. Let R_Y and R_Y be the rank of elements in X and Y respectively. If X has n elements, there are $\frac{n(n-1)}{2}$ pairs of elements in total. Let n_c be the number of pairs that are in the same order in both R_X and R_Y. Also let n_d be the number of pairs that are in apposite order in R_X and R_Y. The Kendall-τ correlation between

R_X and R_Y is calculated as:

$$\tau_{(R_X,R_Y)} = \frac{2(n_c - n_d)}{n(n-1)}$$

Appendix E

Experimental Data

Experiments were conducted on three experimental data: namely (*i*) TREC tracks, (*ii*) the INEX Books Search track, and (*iii*) a Web dataset. We explain each test collection in details.

TREC Tracks

National Institutes of Technology (NIST) organizes a TREC workshop every year. A TREC workshop consists of a set of tracks. Each track focuses on a particular retrieval task, e.g. the Ad-hod retrieval task. The track organizers define the task based on recent challenges in IR community and design an evaluation methodology to support research on the task. Several research groups participate in the track to accomplish the task. The research groups are given a document collection and a set of queries (topics in TREC terminology) created by a set of hired assessors. They index the corpus and run the queries through their retrieval systems, producing sets of result sets that are returned to track organizers. The track organizers use the pooling technique to select a subset of documents to be judged by the assessors who create the queries. The assessors judge the pooled documents and create relevance judgments. Finally, the track organizers assess the performance of participating systems and provide a summary of the evaluation results.

The track organizers later release the document corpus and the query set along with the associated relevance judgments as a new test collection. Also, the summary of evaluation results becomes publicly available which can be used as a set of baseline results by other researchers who intend to evaluate their retrieval model by using the same test collection.

We used two TREC tracks in this thesis, namely (i) TREC-8 Ad-hod track, (ii) TREC 2004 Robust track, which will be explained in the following.

TREC-8 Ad-hoc Track

The Ad-hoc retrieval task assesses the performance of a retrieval system that searches a document corpus using a set of queries. Participants use their retrieval systems to run the queries against the document corpus and return top 1000 documents, retrieved for each query, to NIST to build associated relevance judgments. Participants are free to use any retrieval model to search for the queries. Also, they are allowed to have several runs of their system, each run with a specific setting. Participant can also use any techniques to formulate the queries, e.g. query expansion techniques.

However, the track organizers distinguish among two major categories of query formulation techniques, automatic methods and manual methods. An automatic method is a means of formulating a query with no manual intervention, but a manual method can use human experts to formulate a query. Since manual method require considerably different amount of human effort, care has to be taken when comparing results derived by manual method to the results of automatic ones.

Fifty queries (topics 401-450) were created for the TREC-8 Ad-hod task and the document collection used in this task was the TREC Disks 4 and 5 corpus, excluding the Congressional Record sub-collection.

TREC 2004 Robust Track

The robust retrieval track explores the performance of various retrieval methods by focusing on a particular set of queries known as "poorly performing queries". A query is known as poorly performing if it is not easy for retrieval systems to return its relevant documents. As such, systems' performance on poorly performing queries are usually lower than systems' performance on other queries. The retrieval task in the track is the Ad-hoc retrieval task where the evaluation methodology emphasizes a system's performance on poorly performing queries.

This track uses TREC disks 4 and 5, minus the Congressional Record as the document corpus. Also the query set consists of 49 queries newly created for the task (the 50th was dropped because no relevant documents were found), but also the 50 queries from the TREC 2003 Robust track, and 150 queries from the Ad-hoc tracks of TREC-6 through TREC-8 (1997, 1998, and 1999). Relevance judgments are created for the new queries. Also, relevance judgments created in the earlier tracks are reused for the other queries. Finally, a variant of the mean average precision (MAP) measure, called GMAP [Rob06] that uses a geometric mean rather than an arithmetic mean, is used to measure the systems' average performance by emphasizing on poorly performing queries.

INEX Tracks

Similar to TREC, INEX is a workshop that provides test collections for IR community. The main goal of INEX is to promote the evaluation of focused retrieval by providing large test collections of structured documents, standard evaluation metrics, and a forum for organizations to compare their retrieval systems. Focused Retrieval consists of several tasks including Element Retrieval from an XML document, Page Retrieval from books, as well as Question Answering.[1] INEX organizers usually use the same evaluate methodology, including the same pooling techniques and the rather same evaluate metrics, as being used in TREC community.

INEX Books Search Track

The goal of the Books Search track is to investigate retrieval methods to support users in searching and navigating the full texts of digitized books. The Book Search track in 2010 focused on the following subtasks:[2]

[1] https://inex.mmci.uni-saarland.de/about.html
[2] http://www.inex.otago.ac.nz/tracks/books/books.asp

- Prove It task: the task was to find evidence in books to confirm or refute a fact expressed as a query. Participating systems had to search a collection of 50,00 digitalized books that contain evidence regarding the query's statement.
- Best Books for Reference task: the task was to find the most relevant books on the subject of a given query.
- Active Reading task: the goal of this task was to conduct user studies into active reading, i.e. exploring how and why readers use digitalized books in specific scenarios with a focus on eBook usability.
- Structure Extraction task: the task was to build navigation tools for digitized books by constructing a hyperlinked table of contents from OCR text and layout information for a sample of 1,000 books.

A Web Test Collection

A web test collection of a commercial search engine was used in Chapter 6 to assess the performance of the adaptive query selection. The test collection used the whole web as the document collection. The query set comprised a set of 1,000 queries randomly sampled from the search engine's query log. Fifty various runs of a learning to rank system [Liu09], trained with different feature sets, were considered as participating systems (the details of the training phase is out of the scope of this work). For each run, a random sample of $g = \{5, 10, 20, 30 \text{ or } 40\}$ features was selected from a given feature set. The run was then optimized by using the common training set. For each query, the top 5 web pages returned by the runs were pooled for relevance assessment. A set of assessors hired by the search engine company were asked to create relevance judgments for each of the 1,000 queries. In total, 30,000 relevance judgments were collected. Finally, the performance of each run was measured according to precision at position 5 ($P@5$).

Appendix F

Measuring the Variability in Effectiveness

A typical evaluation of a retrieval system involves computing an evaluation metric, e.g. average precision, for each query of a test collection and then using the average of the metric, e.g. mean average precision, to express the overall effectiveness. However, averages do not capture all the important aspects of effectiveness and, used alone, may not be an informative measure of systems' effectiveness. Indeed, in addition to the average, we need to consider the variation of effectiveness across queries. We refer to this variation as the *variability in effectiveness*. We explore how the variance of a metric can be used as a measure of variability. We define a variability metric, and illustrate how the metric can be used in practice.

Introduction

A common practice in a comparative evaluation of IR systems is to create a test collection comprising a document collection, a set of queries and associated relevance judgments, and to then measure effectiveness of retrieval systems. A typical evaluation of a system involves computing an effectiveness metric, e.g. average precision (AP), and then averaging across queries, e.g. computing the mean average precision (MAP), to characterize the overall system effectiveness. However, when used alone, averages do not capture all the important aspects of effectiveness. For example, averages may not reveal possibly large variations in effectiveness across queries. We maintain that, in addition to average effectiveness, one needs to consider the variation in effectiveness across queries. In particular, when two systems are not distinguishable based on their average, we can use the variations to contrast them. We refer to the cross-query variation as the *variability in effectiveness*.

There are various ways in which variability could be measured. We explore how the variance of IR metrics, in particular, the variance in AP scores, can be used for this purpose. The IR community is, of course, familiar with variance, and uses it routinely to assess whether the difference in the averages of two systems' effectiveness is significant or not. However, our use of variance is different, and we illustrate this next.

Consider a scenario illustrated in Figure F.1a, in which we have two systems, A and B, each of which exhibits the same MAP score, but the variance of AP scores for System A is much larger than for System B. If the two systems are compared based on MAP alone, then a paired student t-test will

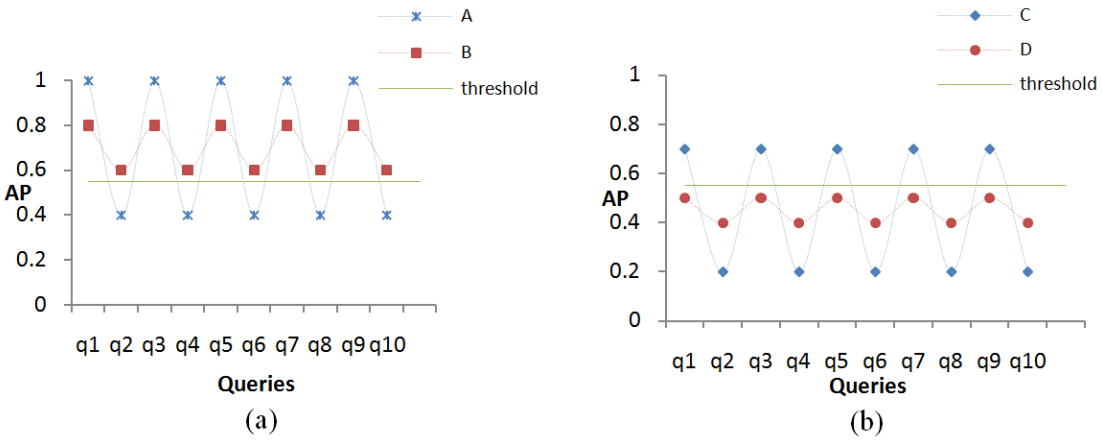


Figure F.1: Two IR systems with (a) equal MAP which is larger than a threshold needed to satisfy a user, and (b) two IR systems with equal MAP smaller than the threshold.

conclude that the two systems are equivalent. However, in practice, users may observe a significant difference between the two systems. Qualitatively, Figure F.1a shows that System A either gives very good or very poor responses to a query. In contrast, System B gives ”satisfactory” responses to all queries, i.e. the responses of System B are neither very good nor very bad. Which system would a user prefer? The answer to this question is not entirely straightforward.

Consider a scenario in which users require AP scores to exceed a minimum threshold in order to be satisfied with the response of the system. This is depicted by the horizontal line in Figure F.1a. In this case, System B always satisfies users, while half the time, System A fails to satisfy users, despite the fact that both systems have the same MAP. In this case, the system with lower variability is preferred. Now consider Figure F.1b, in which we again have two systems, C and D, with the same MAP score. However, in this scenario, the MAP is lower than the threshold needed to satisfy users. In this case, the system with lower variance, D, never satisfies users. In contrast, System C, with high variance does satisfy users for some queries. Thus, the system with higher variability is preferred.

This example highlights three important points. First, the average of a metric does not always provide sufficient information with which to judge a system. Second, variability can be used not just for significance testing, but also to characterize systems with similar average performance. Third, a preference for systems with high or low performance variability depends on the relative performance of systems in comparison with a user’s satisfaction threshold.

Of course, this is an artificial example. However it is common for real systems to exhibit statistically identical mean performance, yet exhibit different levels of variance. For example, TableF.1 shows two experimental runs from the Robust track of TREC 2004 [Voo05]. For each of them we compute MAP across 199 queries (351-450 and 600-700), removing one query that had no relevant documents in the collection. We also calculate the standard deviation of AP scores as our measure of variability. The two runs have the same MAP value but different standard deviations. Using the paired t-test reveals that there is no significant difference in the MAP values while applying a statistical test to assess the equality of standard deviation confirms that the difference in the standard deviations is statistically significant.

Table F.1: Two experimental runs from the robust track of TREC 2004. The corresponding MAP values and standard deviations of AP scores, SD (AP), are measured over 199 queries.

Runs	MAP	SD(AP)	Paired t-test	Levene's test
uogRobSWR5	0.304	0.24	$p = 0.96 \gg 0.05$	$p = 0.007 \ll 0.05$
NLPR04clus10	0.304	0.20		

Background

We first briefly discuss related work and then introduce two statistical significant tests used to compare variabilities in systems' effectiveness.

Related Work

The topic of variability in effectiveness has received little attention in IR research. Perhaps, the most of prior work related to variability is to do query expansion. Query expansion methods typically yield good improvements in mean average precision but are unstable and have high variance across queries [CTC07]. Collins-Thompson [CT09] proposed a model of evaluating effectiveness of query expansion methods by using a risk-reward tradeoff where reward was defined as the percentage gain for MAP relative to the original, un-expanded query, and the risk reflected the number of relevant documents that were lost due to the expansion. Such a risk measurement is solely based on the number of relevant documents. In contrast, the percentage MAP gain depends on not only the number of relevant documents retrieved but also the ranks of them. Perhaps the variability in effectiveness can be an alternative measure of risk where both number of relevant documents and corresponding ranks are taken into account.

C.T. Lee et al. [LVMR$^+$09] proposed a novel weighted average (generalized adaptive-weight mean) to rank systems' effectiveness where the weights reflected the ability of the test topics to differentiate among the retrieval systems. The variance of the AP scores was indirectly incorporated into measuring systems' effectiveness. In particular, they used the Euclidean distance to characterize the dispersion of AP scores. However, effectiveness of their system ranking and comparison was not evaluated in detail. We observe that the performance scores (AP values) are bounded in [0, 1] and expect that the approach will be affected by the boundary conditions, 0 and 1. We propose a way to overcome the issues of a bounded score distribution and its effect on the variance.

Statistical Significance Tests

Tests of statistical significance have been thoroughly discussed in the IR literature. The common statistical significance tests used in IR experiments are student's paired t-test, wilcoxon signed rank, and sign test. The assumptions which these tests are based on were discussed in [Hul93]. In addition, the use of two sampling-based tests, bootstrap shift method and fisher's randomization, in IR was discussed in [SAC07]. Sakai [Sak06] also discussed the use of paired bootstrap test in IR which was a combination of the bootstrap shift method and student's t-test.

These tests, for example, make use of the variance of AP scores to determine whether the difference in two MAP scores is statistically significant. Here, we are interested in determining whether the difference in variabilities, as measured by variance or standard deviation, of two systems' effectiveness is

statistically significant. The statistical community has, of course, addressed this and we briefly describe two tests, the F-test and Levene's test.

F-test

This test first defines a ratio of the standard deviations of two systems' effectiveness measured across a set of queries. Therefore, if σ_A and σ_B are the standard deviations of AP scores of systems A and B, the ratio is calculated as:

$$F = \frac{\sigma_A}{\sigma_B} \tag{F.1}$$

In the F-test the null hypothesis and the alternative hypothesis is defined as below:

$$H_0 : \sigma_A = \sigma_B \text{ (the null hypothesis)}$$

$$H_1 : \sigma_A \neq \sigma_B \text{ (the alternative hypothesis)}$$

The more the ratio deviates from 1, the stronger the evidence for unequal variances. The null hypothesis is rejected if the ratio was larger than a critical value. The critical value is adjusted based on a significance level, e.g. $\alpha = 0.05$ or $\alpha = 0.01$.

There is a limiting condition in F-test assuming that the distribution of AP scores is normal. However, this assumption may not be true in practice. In order to deal with this restriction, we also consider the Levene's test which does not have such an assumption.

Levene's Test

Levene's test is used to assess whether k sample groups have the same standard deviation [Lev60]. Levene's test does not have the normality assumption. The statistic is obtained from one-way analysis of variance (ANOVA), where each observation, in our case each AP score, is replaced with its absolute deviation from the associated group's mean. In our case the group mean is the MAP value. Let $z_{ij} = |AP_{ij} - MAP_i|$, where AP_{ij} is the measured AP value of the i^{th} system on the j^{th} query. Levene's test defines a ratio as:

$$W_0 = \frac{\sum_i n_i(\bar{z}_i - \bar{z})^2 \times \sum_i (n_i - 1)}{(g-1) \times (\sum_i \sum_j (z_{ij} - \bar{z}_i)^2)} \tag{F.2}$$

where g is the number of sample groups which in our case is 2 indicating the number of systems, and n_i is the number of observations in the i^{th} group (in our case it is equal to the number of queries):

$$\bar{z}_i = \frac{\sum z_{ij}}{n_i} \text{ and } \bar{z} = \frac{\sum\sum z_{ij}}{\sum n_i}$$

The null hypothesis is rejected if W_0 was larger than a critical value that is adjusted with regard to a significance level. Replacing the group mean, MAP_{ij}, with the median of observations, median(AP), in forming z_{ij} defines W_{50}. We use W_{50} instead of W_0 when the AP distributions suffer from skewness.

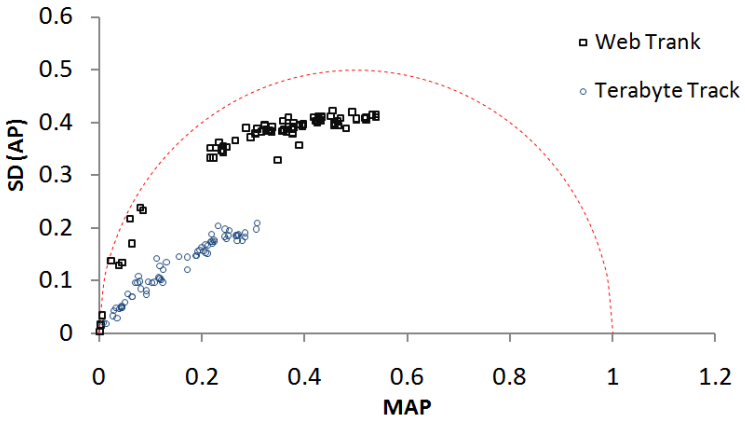


Figure F.2: The standard deviation of AP values (SD (AP)) versus MAP. The standard deviation is bounded in a semicircle with center (0.5, 0.0) and radius 0.5.

Experiments

We examine the performance of various systems involved in the Web and Terabyte tracks of TREC 2004. This study reveals a curious phenomenon - systems with average performances, measured by a bounded IR metric, e.g. MAP, near to 0.5 have larger variances than systems with average performances near to each of the two boundaries, 0 and 1. This phenomenon is an artifact of the fact that the metric's scores are bounded in [0,1].

We proposes two transformations of the metric's scores in order to eliminate this artifact. We then considers all pairs of systems participating in two test collections of TREC 2004: the Web and Robust tracks. Student t-tests show that 26% and 28% of pairs, respectively, are ties, i.e. there is no statistically significant difference in the averages of transformed AP scores. If the variability in effectiveness of these ties is examined, then the F-test shows that 33% and 34% of ties have statistically significant differences in variance. When Levene's test is used, 47% and 38% of ties have statistically significant differences in variance. Finally, We explore the effect that the size of a query set has on the system comparison using variability in effectiveness. We observe that one needs to consider a sample of 90 queries to obtain an error rate smaller than 0.05.

The Variance of a Bounded Metric

Figure F.2 plots the standard deviation in AP scores as a function of MAP, for systems participating in the Web and Terabyte tracks of TREC 2004. The Web track involves 74 systems and 225 queries. The Terabyte track involves 70 systems and 49 queries.

We note that some systems have similar MAP values. Thus, it would be beneficial to use additional criteria, e.g., variability in effectiveness, to differentiate their performance. This is discussed shortly. However, the most striking feature in Figure F.2 is an unexpected trend: the monotonic relationship between standard deviation and MAP, i.e. the larger the MAP value, the larger the variance in AP scores.

We believe this relationship is due to the bounded nature of AP metric, i.e. the fact that the metric's values fall within [0,1]. This bounds the standard deviation of AP scores to a semicircle as shown in

Figure F.2 and proven as below.

Lemma: For all data sets like $X = \{x_1, x_2, ..., x_N\}$ where $0 \leq x_i \leq 1$, the corresponding *mean-standard deviation* values, $(\bar{X}, S_x)$, are confined within a semicircle with center (0.5, 0) and radius r=0.5:

$$(\bar{X} - \frac{1}{2})^2 + S_x^2 \leq (\frac{1}{2})^2;$$

$$\bar{X}^2 + S_x^2 \leq \bar{X} \tag{F.3}$$

Proof: with reference to the mean and variance:

$$\bar{X} = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{F.4}$$

$$S_x^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{X})^2 = \frac{1}{N}\sum_{i=1}^{N} x_i^2 - 2 \times \bar{X}(\frac{1}{N}\sum_{i=1}^{N} x_i) + \bar{X}^2 = \frac{1}{N}\sum_{i=1}^{N} x_i^2 - \bar{X}^2$$

therefore:

$$\bar{X}^2 + S_x^2 = \frac{1}{N}\sum_{i=1}^{N} x_i^2 \tag{F.5}$$

$x_i^2 \leq x_i$ because $0 \leq x_i \leq 1$; therefore:

$$\frac{1}{N}\sum_{i=1}^{N} x_i^2 \leq \frac{1}{N}\sum_{i=1}^{N} x_i = \bar{X} \tag{F.6}$$

considering (F.5) and (F.6) together:

$$\bar{X}^2 + S_x^2 = \frac{1}{N}\sum_{i=1}^{N} x_i^2 \leq \frac{1}{N}\sum_{i=1}^{N} x_i = \bar{X}$$

then we reach to (F.3):

$$\bar{X}^2 + S_x^2 \leq \bar{X}$$

Therefore, the retrieval system with MAP close to one of the boundaries, 0 or 1, are more likely to have a smaller variance than those with MAP near to the center, 0.5. For this reason using the standard deviation of the raw AP scores is not a reliable measure of variability. In fact, this is true for any other bounded IR metrics, e.g. the reciprocal rank, as shown in Figure F.3. The figure shows how the variance decreases as the MRR increases above 0.5. Again, this is expected since now the variation above the mean is limited by the upper bound of one on reciprocal rank.

In order to overcome this issue, we consider functions that map values from [0,1] to $(-\infty, +\infty)$. We favor mappings that produce a symmetric distribution in the transformed space, akin to the normal distribution, if possible. We can then define the variability in effectiveness as the variance of the transformed values of a metric.

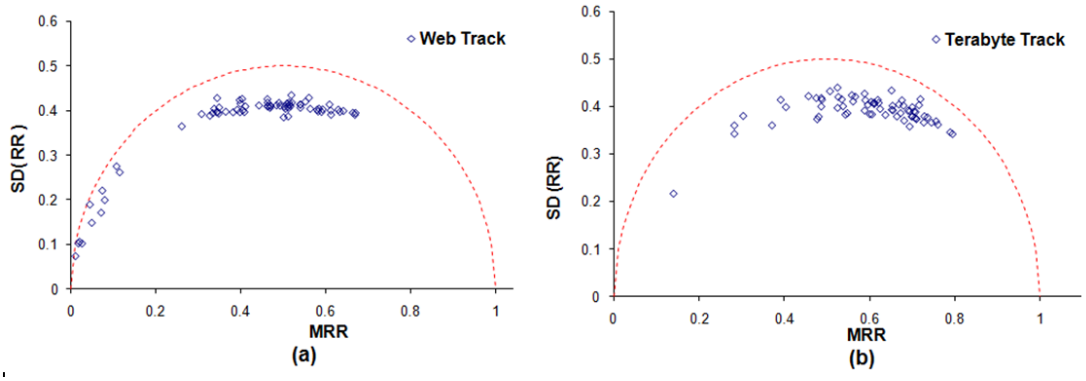


Figure F.3: MRR versus the standard deviation of RR values from: (a) runs participating in the Web track 2004, (b) runs participating in the Terabyte track 2004.

The Variability of Transformed Scores

We illustrate our approach by considering two transformations that have been used in IR and observe the properties of the transformed scores. The first is the *logit* function used by Cormak and Lynam [CL06] as a parametric estimate to deal with the asymmetric AP distribution. The logit is defined as: $logit(x) = log(\frac{x}{1-x})$ for x in (0,1). The boundary points, 0 and 1, are replaced by ϵ and $1-\epsilon$ respectively, for a small value of $\epsilon > 0$, and then transformed, letting $\epsilon \to 0$.

The second transformation is the *standardized score* or *z-score* whose use in IR was motivated by Webber et al. [WMZ08a]. It is defined as $z = \frac{(x-\bar{x})}{\sigma}$, where x is a metric's score, e.g. an AP score. In addition, $\bar{x}$ and σ are the average and standard deviation of a set of scores measured across a set of retrieval systems on a fixed query. Hence, for a particular query it is defined as

$$z = \frac{(AP - Mean(AP)_{systems})}{SD(AP)_{systems}} \tag{F.7}$$

In the following, we observe several properties of the logit(AP) and the standardized z-scores.

Boundary Values and Score Distributions

Figure F.4 shows three runs of the Web track collection: (a) with a low MAP value, (b) with a medium MAP value, and (c) with a high MAP value. The distributions of the AP scores before and after both the logit and z-score transformations are presented as frequency histograms. The logit and z-score transformations differ significantly in the way they handle boundary values. The logit transformation transforms the boundaries to extreme values in the transformed space. This is observed by the extreme values at each end of the distributions in the middle column. In contrast, the z-score transformation disperses the boundaries smoothly as illustrated in the right column. In addition, the z-score transformation helps eliminate the source of variance coming from query difficulty[1] before measuring the variability of system effectiveness itself.

We now consider the variability in a system's effectiveness as the standard deviation of the transformed AP values. Let MLAP refer to the mean of the logit-transformed AP values and let MSAP refer to the mean of the standardized z-transformed AP values. Figure F.5 shows the scatterplots of the stan-

[1] A query is regarded as difficult if the range of effectiveness scores measured across a set of systems is small and near to zero.

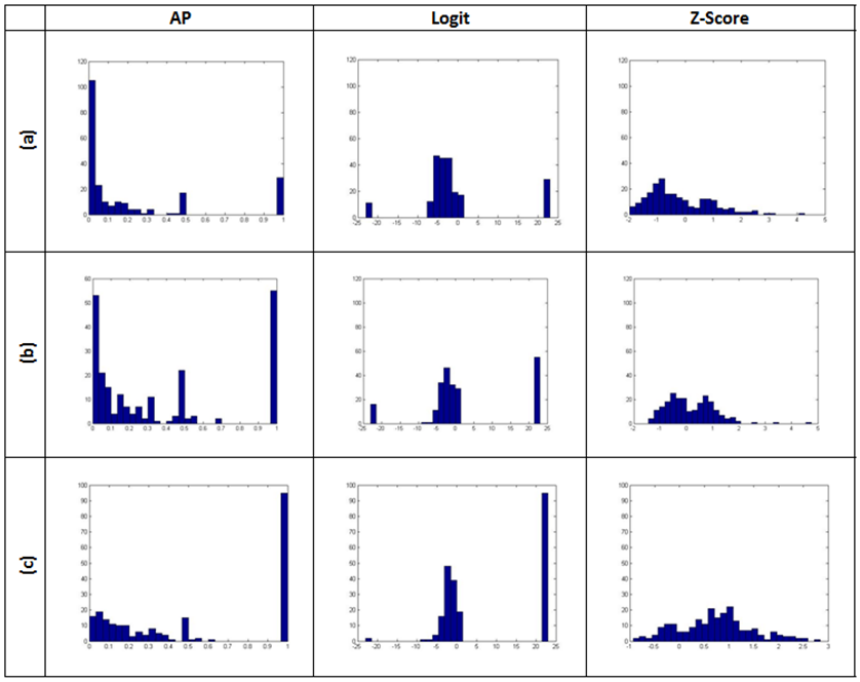


Figure F.4: The frequency distributions before and after transformation of three runs in Web track. (a) a run with a low MAP, (b) a run with a medium MAP, (c) a run with a high MAP.

dard deviations in transformed AP values as a function of their mean values, MLAP and MSAP. As seen in the figure, the logit and z-score transform the scores in different ranges. In addition, there is no longer a monotonic relationship between the values of mean and variability.

Variability as a Tie Breaker

We consider all pairs of the top 75% (ordered by MAP) of systems participating in either the Robust or Web track of TREC 2004. We compare systems based on the mean of the standardized z-scores (MSAP). We use the paired t-test to measure the significance of MSAP differences. We set the significance level to 0.05. For all the ties we use the F-test and Levene's test to investigate the proportion of ties for which the variabilities in effectiveness's scores are significantly different.

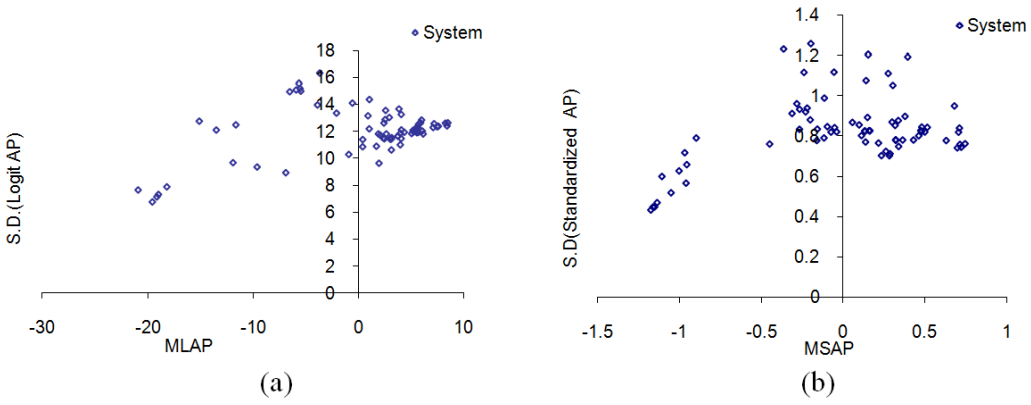


Figure F.5: Variability in effectiveness versus mean of transformed AP values: logit (a) and the z-score transformation (b).

Table F.2: The variability in effectiveness as a tie breaker: number of pairs, ties and broken ties in two tracks of TREC 2004.

Collections	Pairs	Status	Ties	Broken ties	
				F-test	Levene
Robust	3321	before transformation	997 (30%)	0 (0%)	106 (11%)
		after transformation	857 (26%)	280 (33%)	404 (47%)
Web	1485	before transformation	469 (31%)	1 (0.002%)	21 (0.04%)
		after transformation	415 (28%)	140 (34%)	158 (38%)

```
foreach query set size c from 10, 20, 30, ... , 100 {
  set the counters to 0;
   foreach TREC test collection t {
     foreach pair of systems A and B from track t {
       foreach trial from 1 to 50
         select two disjoint sets of queries X and Y of size c from t;
          if ( the difference between the variabilities is significant){
              d_X=SD(A,X)-SD(B,X);
              d_Y=SD(A,Y)-SD(B,Y);
              increment counter;
              if(d_X * d_Y < 0) {
                   increment swap counter; } } }}
     error-rate (c) =swap counter /counter;}
```

Figure F.6: Calculating error rates. SD(A, X) is the standard deviation of AP scores of system A measured on the query Set X.

As seen in Table F.2 for the Robust track, 30% of pairs are considered ties, when using AP score, and 26% are considered ties in the transformed space. Interestingly, before transformation, the F-test cannot distinguish any statistical difference in variability, and the Levene's test can only break 11% of the ties. In contrast, after transformation into the z-space, the F-test can distinguish between 33% and Levene's test can distinguish between 47% of the tied pairs. A similar effect before and after transformation is observed for the Web track.

The Effect of Query Set Size on Measuring Variability in Effectiveness

If we are to use variability to characterize systems, it will be useful to know how many queries are needed to reliably compare two systems in terms of variability in effectiveness. Indeed, we will need to know how likely a decision would change if we compare systems using a different query set. This performance variation across query sets has previously been studied in the context of average performance [VB02]. We perform the same experiment to compare variabilities in systems' effectiveness.

In our experiment, we first fix the query set size, and then compute the variabilities in effectiveness of a pair of systems, A and B. Let us assume that System A is less volatile than System B based on this measurement. We then estimate the probability of a changed decision, i.e. finding System B to be less volatile than System A. We estimate this probability by comparing the two systems across several trials that use different query sets and then counting how many times the preference decision changes. Finally, to estimate the average probability of changing a decision, we repeat the process on different pairs of systems. This average probability (across systems) is called the *error rate*. The whole process

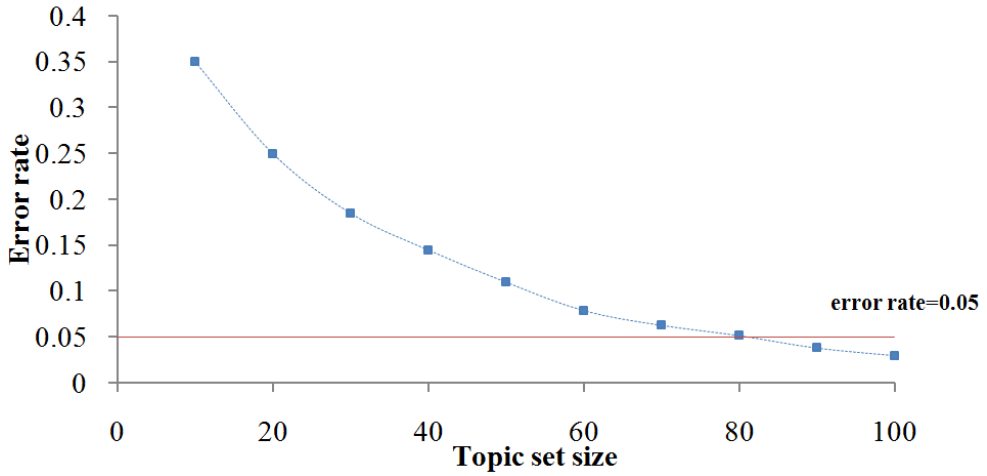


Figure F.7: Error rate versus query set size using two TREC test collections: web track and robust track of TREC 2004.

is repeated for different sizes of query sets.

The algorithm for computing the error rate is shown in Figure F.6. It is based on the algorithm described in [VB02]. In our experiment, we compute the error rate for all the pairs regardless of their absolute differences. We run 50 different trials using different combinations of queries in the two disjoint query sets. Furthermore, as Sanderson and Zobel [SZ05] suggested, we only consider pairs with statistically significant differences in variability, as measured by Levene's test with a significance level of 0.05.

Once again we use the runs participating in the Robust track of TREC 2004 using queries 351-450 and 601-700 (199 queries), and the runs participating in the Web track (225 queries). In this experiment, only the top 75% of systems (ranked by MAP) are considered to prevent the poorly performing runs from having an effect on our conclusion [VB02]. Thus, our data collection consists of 135 runs and 4806 pairs of runs. Note that we transform AP scores using the z-score before measuring variability. The resulting error rate is shown in Figure F.7. As expected, the curve shows that the error rate decreases as the query set size increases. The experiment indicates that 90 queries are required to obtain an error rate less than 0.05. With 80 queries the measured error rate is 0.052 and with 90 queries it is 0.038.

Conclusion

The average of effectiveness, measured across a query set, does not capture all the important aspects of effectiveness and, used alone, may not be an informative measure of a system's effectiveness. We defined variability in effectiveness as the standard deviation of effectiveness scores measured across a set of queries. We proposed that a mean-variance graph helps demonstrate effectiveness in a two-dimensional space rather than ranking systems based on their average effectiveness. Our investigation revealed that the bounded values of a metric yield a curious phenomenon where values of average around 0.5 are accompanied with higher variances. We attributed this to the fact that the metric values fall within [0, 1]. This bounds the standard deviation of the scores to a semicircle. Hence, retrieval systems

with average effectiveness close to each of the two boundaries have smaller variances than those with average away from the boundaries. However, there might be also other reasons. For example, when the distribution is not symmetric, standard deviation cannot explain the dispersion properly. In Figure F.3 it was shown that the distributions of AP scores were skewed toward the upper boundary, 1, and was completely asymmetric. We used two transformation methods to deal with this problem and showed how they differentiate systems effectiveness with the same average score. We finally discussed the minimum sample size required to estimate the variability in effectiveness. In our experiments we observed that 90 queries were required to obtain an error rate less than 0.05.

We only considered standard deviation as the measure of variability while it would be interesting to consider other measures, e.g. interquartile range and median absolute deviation. In addition, there are several ways to transform scores in a more symmetric space. For example, one might consider both logit and z-score transformation together. That is, the AP scores are first transformed by logit to ($-\infty$, $+\infty$) and then z-score is used to deal with extreme values. As truly shown by Lin and Hauptmann [LH05], the minimum sample size varies across pairs of systems, and it depends on the difference between two systems' average effectiveness scores and corresponding variances.

Mean and variability can be used to evaluate retrieval systems. One may define a new metric as a function of both mean and variability. Such a metric helps rank systems' effectiveness in a one-dimensional space by considering both mean and variability in effectiveness. In addition, by a hypothetical scenario we showed that how a threshold of user satisfaction helps make preference between volatile and stable systems. However, we need to at least deal with two issues here. Firstly, in order to measure users' satisfaction we need to evaluate systems from users' perspective, i.e. directly asking users to express the amount of satisfaction. Such a user-oriented evaluation method provides accurate results but it is extremely expensive and difficult to do correctly. We can also model users' satisfaction by using implicit feedbacks of users, e.g. click-through data in a search engine query log. This method is less expensive but inaccurate. Secondly, users' satisfaction threshold may vary across queries. Indeed, the scenario described in the introduction section was simplified by considering the threshold as a constant value. However, in practice, the threshold varies across queries since it is highly depended on users' information needs and their expectation of the result set. We will consider these issues for future work.

Appendix G

Publications

[1] M.Hosseini, I.J.Cox, N.Milic-Frayling, M.Shokouhi, and E.Yilmaz. An Uncertainty-aware Query Selection for Evaluation of IR Systems. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in Information Retrieval*, SIGIR'12, pages 901-910, Oregon, Portland, USA, 2012. ACM.

[2] M.Hosseini, I.J.Cox, N.Milic-Frayling, G.Kazai, and V.Vinay. On Aggregating Labels from Multiple Crowd Workers to Infer Relevance of Documents. In *Proceedings of the 34th European conference on Advances in information retrieval*, ECIR'12, pages 182-194, Barcelona, Spain, Heidelberg, 2012. Springer-Verlag.

[3] M. Hosseini, I. J. Cox, N. Milic-Frayling, T. Sweeting, and V. Vinay. Prioritizing relevance judgments to improve the construction of IR test collections. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, CIKM'11, pages 641-646, Glasgow, UK, 2011. ACM.

[4] M. Hosseini, I. J. Cox, N. Milic-Frayling, V. Vinay, and T. Sweeting. Selecting a subset of queries for acquisition of further relevance judgements. In *Proceedings of the Third international conference on Advances in information retrieval theory*, ICTIR'11, pages 113-124, Berlin, Heidelberg, 2011. Springer-Verlag.

[5] M. Hosseini, I. J. Cox, and N. Milic-Frayling. Optimizing the cost of information retrieval test collections. In *Proceedings of the 4th workshop on Workshop for Ph.D. students in conjunction with the 20th ACM international conference on Information and knowledge management*, CIKM'11, pages 79-82, Glasgow, UK, 2011. ACM.

[6] M. Hosseini, I. J. Cox, N. Milic-Frayling, and V. Vinay. Measuring the variability in effectiveness of a retrieval system. In *Proceedings of the First international Information Retrieval Facility conference on Advances in Multidisciplinary Retrieval*, IRFC'10, pages 70-83, Berlin, Heidelberg, 2011. Springer-Verlag.

[7] M.Hosseini. A study on performance volatility in information retrieval. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR'09, 853–853, Boston, MA, USA, 2009. ACM.

Bibliography

[ACA+07] James Allan, Ben Carterette, Javed A. Aslam, Virgil Pavlu, Blagovest Dachev, and Evangelos Kanoulas. TREC 2007 million query track. In *Notebook Proceedings of TREC 2007*. TREC, 2007.

[Alo11] Omar Alonso. Crowdsourcing for information retrieval experimentation and evaluation. In *Proceedings of the Second international conference on Multilingual and multimodal information access evaluation*, CLEF'11, pages 2–2, Berlin, Heidelberg, 2011. Springer-Verlag.

[AM09] Omar Alonso and Stefano Mizzaro. Can we get rid of trec assessors? using mechanical turk for relevance assessment. In *Proceedings of of the 32nd annual international ACM SIGIR conference on Research and development in information retrieval,: Workshop on The Future of IR Evaluation*, 2009.

[AP08] Javed A. Aslam and Virgil Pavlu. A practical sampling strategy for efficient retrieval evaluation. In *Technical report, Computer Science Department,North Eastern University*, pages 1–10, 2008.

[APY06] Javed A. Aslam, Virgil Pavlu, and Emine Yilmaz. A statistical method for system evaluation using incomplete judgments. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 541–548, New York, NY, USA, 2006. ACM.

[BCS+08] Peter Bailey, Nick Craswell, Ian Soboroff, Paul Thomas, Arjen P. de Vries, and Emine Yilmaz. Relevance assessment: are judges exchangeable and does it matter. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pages 667–674, New York, NY, USA, 2008. ACM.

[BCYS07] Stefan Büttcher, Charles L. A. Clarke, Peter C. K. Yeung, and Ian Soboroff. Reliable information retrieval evaluation with incomplete and biased judgements. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 63–70, New York, NY, USA, 2007. ACM.

[Bil95] Patrick Billingsley. *Probability and Measure*. New York: Wiley, New York, NY, USA, 1995.

[BV00] Chris Buckley and Ellen M. Voorhees. Evaluating evaluation measure stability. In *SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 33–40, New York, NY, USA, 2000. ACM.

[BV04a] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

[BV04b] Chris Buckley and Ellen M. Voorhees. Retrieval evaluation with incomplete information. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 25–32, New York, NY, USA, 2004. ACM.

[CA05] Ben Carterette and James Allan. Incremental test collections. In *CIKM '05: Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 680–687, New York, NY, USA, 2005. ACM.

[CAS06] Ben Carterette, James Allan, and Ramesh Sitaraman. Minimal test collections for retrieval evaluation. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 268–275, New York, NY, USA, 2006. ACM.

[CGJM10] Ben Carterette, Evgeniy Gabrilovich, Vanja Josifovski, and Donald Metzler. Measuring the reusability of test collections. In *WSDM '10: Proceedings of the third ACM international conference on Web search and data mining*, pages 231–240, New York, NY, USA, 2010. ACM.

[CK67] C A Cuadra and R V Katter. Opening the black box of relevance. *Journal of Documentation*, 23(4):291–303, 1967.

[CKPF10] Ben Carterette, Evangelos Kanoulas, Virgil Pavlu, and Hui Fang. Reusable test collections through experimental design. In *SIGIR '10: Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 547–554, New York, NY, USA, 2010. ACM.

[CL06] Gordon V. Cormack and Thomas R. Lynam. Statistical precision of information retrieval evaluation. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 533–540, New York, NY, USA, 2006. ACM.

[CM97] Cyril W. Cleverdon and J. Mills. The testing of index language devices. pages 98–110, 1997.

[Coo68] W.S. Cooper. Expected search length: A single measure of retrieval effectiveness based on the weak ordering action of retrieval systems. *American Documentation*, 19:30–41, 1968.

[CPC98] Gordon V. Cormack, Christopher R. Palmer, and Charles L. A. Clarke. Efficient construction of large test collections. In *SIGIR '98: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 282–289, New York, NY, USA, 1998. ACM.

[CPK+08] Ben Carterette, Virgil Pavlu, Evangelos Kanoulas, Javed A. Aslam, and James Allan. Evaluation over thousands of queries. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 651–658, New York, NY, USA, 2008. ACM.

[CS07] Ben Carterette and Mark D. Smucker. Hypothesis testing with incomplete relevance judgments. In *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 643–652, New York, NY, USA, 2007. ACM.

[CS10] Ben Carterette and Ian Soboroff. The effect of assessor error on ir system evaluation. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '10, pages 539–546, New York, NY, USA, 2010. ACM.

[CT09] Kevyn Collins-Thompson. Robust word similarity estimation using perturbation kernels. In *ICTIR '09: Proceedings of the 2nd International Conference on Theory of Information Retrieval*, pages 265–272, Berlin, Heidelberg, 2009. Springer-Verlag.

[CTC07] Kevyn Collins-Thompson and Jamie Callan. Estimation and use of uncertainty in pseudo-relevance feedback. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 303–310, New York, NY, USA, 2007. ACM.

[CV95] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Mach. Learn.*, 20(3):273–297, September 1995.

[DGH52] D J Thompson D G Horvitz. A generalization of sampling without replacement from a finite universe. *Journal of American Statistical Association*, 47(260):663–685, 1952.

[DS79] A P Dawid and A M Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. *Journal of the Royal Statistical Society Series C Applied Statistics*, 28(1):20–28, 1979.

[EHP00] M. Evans, N.A.J. Hastings, and J.B. Peacock. *Statistical Distributions*. Wiley Series in Probability and Statistics. Wiley, 2000.

[GMR09] John Guiver, Stefano Mizzaro, and Stephen Robertson. A few good topics: Experiments in topic set reduction for retrieval evaluation. *ACM Trans. Inf. Syst.*, 27(4), 2009.

[Har95] Donna Harman. Overview of the second text retrieval conference (trec-2). *Inf. Process. Manage.*, 31(3):271–289, 1995.

[HCMF+11] Mehdi Hosseini, Ingemar J. Cox, Natasa Milic-Frayling, Vishwa Vinay, and Trevor Sweeting. Selecting a subset of queries for acquisition of further relevance judgements. In *Proceedings of the Third international conference on Advances in information retrieval theory*, ICTIR'11, pages 113–124, Berlin, Heidelberg, 2011. Springer-Verlag.

[HHdJA09] Claudia Hauff, Djoerd Hiemstra, Franciska de Jong, and Leif Azzopardi. Relying on topic subsets for system ranking estimation. In *Proceedings of the 18th ACM conference on Information and knowledge management*, CIKM '09, pages 1859–1862, New York, NY, USA, 2009. ACM.

[Hub74] Peter Huber. *Robust Statistics*. Wiley, New York, 1974.

[Hul93] David Hull. Using statistical testing in the evaluation of retrieval experiments. In *SIGIR '93: Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 329–338, New York, NY, USA, 1993. ACM.

[JK02] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, 2002.

[KKKMF11] Gabriella Kazai, Jaap Kamps, Marijn Koolen, and Natasa Milic-Frayling. Crowdsourcing for book search evaluation: impact of hit design on comparative system ranking. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, SIGIR '11, pages 205–214, New York, NY, USA, 2011. ACM.

[KL11] Abhimanu Kumar and Matthew Lease. Modeling annotator accuracies for supervised learning. In *Proceedings of of the 4st ACM internation conference on Web Search and Data Mining: Workshop on Crowdsourcing for Search and Data Mining*, page 19, 2011.

[Koh95a] Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th international joint conference on*

Artificial intelligence - Volume 2, IJCAI'95, pages 1137–1143, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

[Koh95b] Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th international joint conference on Artificial intelligence - Volume 2*, IJCAI'95, pages 1137–1143, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

[Lev60] H Levene. Robust test for equality of variances. *Contributions to Probability and Statistics: Essays in Honor of Harold Hotteling*, pages 278–292, 1960.

[LH05] Wei-Hao Lin and Alexander Hauptmann. Revisiting the effect of topic set size on retrieval error. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 637–638, New York, NY, USA, 2005. ACM.

[Liu09] Tie-Yan Liu. Learning to rank for information retrieval. *Found. Trends Inf. Retr.*, 3(3):225–331, March 2009.

[LVMR+09] Chung Tong Lee, Vishwa Vinay, Eduarda Mendes Rodrigues, Gabriella Kazai, Nataša Milic-Frayling, and Aleksandar Ignjatovic. Measuring system performance and topic discernment using generalized adaptive-weight mean. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*, pages 2033–2036, New York, NY, USA, 2009. ACM.

[MK60] M. E. Maron and J. L. Kuhns. On relevance, probabilistic indexing and information retrieval. *J. ACM*, 7(3):216–244, 1960.

[MoWMMRCC67] O.L. Mangasarian, University of Wisconsin-Madison. Mathematics Research Center, and WISCONSIN UNIV MADISON MATHEMATICS RESEARCH CENTER. *Nonlinear Fractional Programming*. MRC technical summary report. Mathematics Research Center, University of Wisconsin, 1967.

[MR07] Stefano Mizzaro and Stephen Robertson. Hits hits TREC: exploring IR evaluation results with network analysis. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 479–486, New York, NY, USA, 2007. ACM.

[MRS08] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schtze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[Mur88] K.G. Murty. *Linear complementarity, linear and nonlinear programming*. Sigma series in applied mathematics. Heldermann, 1988.

[MW10] Winter Mason and Duncan J. Watts. Financial incentives and the "performance of crowds". *SIGKDD Explor. Newsl.*, 11(2):100–108, May 2010.

[MZ08] Alistair Moffat and Justin Zobel. Rank-biased precision for measurement of retrieval effectiveness. *ACM Trans. Inf. Syst.*, 27(1):1–27, 2008.

[NR10] Stefanie Nowak and Stefan Rüger. How reliable are annotations via crowdsourcing: a study about inter-annotator agreement for multi-label image annotation. In *Proceedings of the international conference on Multimedia information retrieval*, MIR '10, pages 557–566, New York, NY, USA, 2010. ACM.

[Pla00] John C. Platt. Probabilistic outputs for support vector machines and comparison to regularized likelihood methods. In *Advances in Large Margin Classifiers*, 2000.

[Rob97] Stephen E. Robertson. The probability ranking principle in IR. pages 281–286, 1997.

[Rob06] Stephen Robertson. On gmap: and other transformations. In *CIKM '06: Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 78–83, New York, NY, USA, 2006. ACM.

[Rob11] Stephen Robertson. On the contributions of topics to system evaluation. In *Proceedings of the 33rd European conference on Advances in information retrieval*, ECIR'11, pages 129–140, Berlin, Heidelberg, 2011. Springer-Verlag.

[SAC07] Mark D. Smucker, James Allan, and Ben Carterette. A comparison of statistical significance tests for information retrieval evaluation. In *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 623–632, New York, NY, USA, 2007. ACM.

[Sak06] Tetsuya Sakai. Evaluating evaluation metrics based on the bootstrap. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 525–532, New York, NY, USA, 2006. ACM.

[Sak07] Tetsuya Sakai. Alternatives to bpref. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 71–78, New York, NY, USA, 2007. ACM.

[Sak08] Tetsuya Sakai. Comparing metrics across trec and ntcir: the robustness to system bias. In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management*, pages 581–590, New York, NY, USA, 2008. ACM.

[SFR07] Mark Schmidt, Glenn Fung, and Rómer Rosales. Fast optimization methods for l1 regularization: A comparative study and two new approaches. In *Proceedings of*

the 18th European conference on Machine Learning, ECML '07, pages 286–297, Berlin, Heidelberg, 2007. Springer-Verlag.

[SJ11a] Mark Smucker and Chandra Prakash Jethani. The crowd vs. the lab: A comparison of crowd-sourced and university laboratory participant behavior. In *Proceedings of of the 34st annual international ACM SIGIR conference on Research and development in information retrieval: Workshop on Crowdsourcing for Information Retrieval*, 2011.

[SJ11b] Mark D. Smucker and Chandra Prakash Jethani. Measuring assessor accuracy: a comparison of nist assessors and user study participants. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, SIGIR '11, pages 1231–1232, New York, NY, USA, 2011. ACM.

[SJvR76] Karen. Sparck Jones and Keith van Rijsbergen. Information retrieval test collections. *Journal of Documentation*, 32(1):59–75, 1976.

[SNC01] Ian Soboroff, Charles Nicholas, and Patrick Cahan. Ranking retrieval systems without relevance judgments. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '01, pages 66–73, New York, NY, USA, 2001. ACM.

[SOJN08] Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 254–263, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[SSR06] R.C. Sprinthall, E. Sprinthall, and Cram101 Textbook Reviews. *Basic Statistical Analysis*. Cram 101. Academic Internet Publishers Incorporated, 2006.

[STS11] Falk Scholer, Andrew Turpin, and Mark Sanderson. Quantifying test collection quality based on the consistency of relevance judgements. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, SIGIR '11, pages 1063–1072, New York, NY, USA, 2011. ACM.

[SZ05] Mark Sanderson and Justin Zobel. Information retrieval system evaluation: effort, sensitivity, and reliability. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 162–169, New York, NY, USA, 2005. ACM.

[VB02] Ellen M. Voorhees and Chris Buckley. The effect of topic set size on retrieval experiment error. In *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR*

conference on Research and development in information retrieval, pages 316–323, New York, NY, USA, 2002. ACM.

[Voo98] Ellen M. Voorhees. Variations in relevance judgments and the measurement of retrieval effectiveness. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '98, pages 315–323, New York, NY, USA, 1998. ACM.

[Voo02] Ellen M. Voorhees. The philosophy of information retrieval evaluation. In *CLEF '01: Revised Papers from the Second Workshop of the Cross-Language Evaluation Forum on Evaluation of Cross-Language Information Retrieval Systems*, pages 355–370, London, UK, 2002. Springer-Verlag.

[Voo05] Ellen M. Voorhees. The trec robust retrieval track. *SIGIR Forum*, 39(1):11–20, 2005.

[WEST03] Jason Weston, André Elisseeff, Bernhard Schölkopf, and Mike Tipping. Use of the zero norm with linear models and kernel methods. *J. Mach. Learn. Res.*, 3:1439–1461, March 2003.

[WMZ08a] William Webber, Alistair Moffat, and Justin Zobel. Score standardization for inter-collection comparison of retrieval systems. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 51–58, New York, NY, USA, 2008. ACM.

[WMZ08b] William Webber, Alistair Moffat, and Justin Zobel. Statistical power in retrieval experimentation. In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management*, pages 571–580, New York, NY, USA, 2008. ACM.

[YA06] Emine Yilmaz and Javed A. Aslam. Estimating average precision with incomplete and imperfect judgments. In *CIKM '06: Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 102–111, New York, NY, USA, 2006. ACM.

[YKA08] Emine Yilmaz, Evangelos Kanoulas, and Javed A. Aslam. A simple and efficient sampling method for estimating ap and ndcg. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 603–610, New York, NY, USA, 2008. ACM.

[Zob98] Justin Zobel. How reliable are the results of large-scale information retrieval experiments? In *SIGIR '98: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 307–314, New York, NY, USA, 1998. ACM.