# Modeling and Solving Term Mismatch for Full-Text Retrieval

Le Zhao

CMU-LTI-13-002

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213
www.lti.cs.cmu.edu

**Thesis Committee:**
Jamie Callan, Chair
Jaime Carbonell
Yiming Yang
Bruce Croft, University of Massachusetts at Amherst

*Submitted in partial fulfillment of the requirements*
*for the degree of Doctor of Philosophy*
*in Language and Information Technologies*



Supported by National Science Foundation grant IIS-1018317. The views and conclusions are the author's, and do not necessarily reflect those of the sponsor.

**Keywords:** Term mismatch, automatic prediction, efficiency, probabilistic retrieval models, query diagnosis, term weighting, query expansion, term expansion, conjunctive normal form queries, user interaction

*for Jasmine and little Gloria*

## Abstract

Even though modern retrieval systems typically use a multitude of features to rank documents, the backbone for search ranking is usually the standard tf.idf retrieval models.

This thesis addresses a limitation of the fundamental retrieval models, the term mismatch problem, which happens when query terms fail to appear in the documents that are relevant to the query. The term mismatch problem is a long standing problem in information retrieval. However, it was not well understood how often term mismatch happens in retrieval, how important it is for retrieval, or how it affects retrieval performance. This thesis answers the above questions, and proposes principled solutions to address this limitation. The new understandings of the retrieval models will benefit its users, as well as inform the development of software applications built on top of them.

This new direction of research is enabled by the formal definition of the probability of term mismatch, and quantitative data analyses around it. In this thesis, term mismatch is defined as the probability of a term not appearing in a document that is relevant to the query. The complement of term mismatch is the term recall, the probability of a term appearing in relevant documents. Even though the term recall probability is known to be a fundamental quantity in the theory of probabilistic information retrieval, prior research in ad hoc retrieval provided few clues about how to estimate term recall reliably.

This dissertation research designs two term mismatch prediction methods. With exploratory data analyses, this research first identifies common reasons that user-specified query terms fail to appear in documents relevant to the query, develops features correlated with each reason, and integrates them into a predictive model that can be trained from data. This prediction model uses training queries with relevance judgments to predict term mismatch for test queries without known relevance, and can be viewed as a type of transfer learning where training queries represent related ranking tasks that are used by the learning algorithm to facilitate the ranking for new test tasks. Further data analyses focus on the variation of the term mismatch probability for the same term across different queries, and demonstrate that query dependent features are needed for effective term mismatch prediction. At the same time, because the cross-query variation of term mismatch is small for most of the repeating term occurrences, a second mismatch prediction method is designed to use historic occurrences of the same term to predict the mismatch probability for its test occurrences. This provides an alternative and more efficient procedure to predict term mismatch.

Effective term mismatch predictions can be used in several different ways to improve retrieval. The probabilistic retrieval theory suggests to use the term recall probabilities as term weights in the retrieval models. Experiments on 6 different TREC Ad hoc track and Web track datasets show that this automatic intervention improves both retrieval recall and precision substantially for long queries. Even though term weighting does not substantially improve retrieval accuracy for short queries which typically have a higher baseline performance, much larger gains are possible by solving mismatch using user expanded Conjunctive Normal Form queries. These queries try to fix the mismatch problem by expanding every query term individually. Our method uses the automatic term mismatch predictions as a diagnostic tool to guide interactive interventions, so that the users can expand the query terms that need expansion most. Simulated expansion interactions based on real user-expanded queries on TREC Ad hoc and Legal track datasets show that expanding the terms that have the highest predicted mismatch probabilities effectively improves retrieval performance. The resulting Boolean Conjunctive Normal Form expansion queries are both compact and effective, substantially outperforming the short keyword queries as well as the traditional bag of

word expansion that may use the same set of high quality manual expansion terms.

Promising problems for future research are identified, together with research areas where the term mismatch research may make an impact.

## Acknowledgments

Jamie Callan, my thesis advisor, has been constantly contributing ideas into the research, has given me the freedom to explore, and has provided full support for this research even at the beginning where this research does not seem promising yet and is not well aligned with the original plan of working on structured retrieval. Looking back my last 6 years at Carnegie Mellon, Jamie has provided plenty and careful guidance at the beginning, but has gradually given me more and more freedom to try new ideas. Jamie has given me the opportunity to work on a large number of tasks related to my interest in core retrieval modeling. Jamie's advices and encouragements always come at the appropriate moment and place and with the right amount, keeping me busy and focused. Without Jamie, this journey would seem endless, it would be very easy to get lost and it would not be fun. I am very glad that all our efforts were not wasted. A part of the work has turned into this dissertation, and the rest prepared me well for my future adventures.

Up to the writing of this document, anonymous reviewers from 4 venues (conferences and NSF) contributed lots of helpful suggestions and comments. My officemates Ni Lao and Frank Lin contributed through constant office room discussions. Interactions with lots of other people either face to face or through email has made the work better in various ways, they include Yiming Yang, Nick Craswell, Stephen Robertson, Susan Dumais, Yi-Min Wang, Mandar Mitra, Charlie Clarke, Ellen Voorhees, John Tait, Hui (Grace) Yang, Vitor Carvolho, Siddharth Gopal, Jaime Arguello, Jon Elsas, Matthew Bilotti, Anagha Kulkarni-Joshi, Pinar Donmez, Vamshi Ambati, Ben Carterette, Jin Young Kim, Michael Bendersky and Matthew Lease. My bay area friends provided their full support in helping me hone my talks, including Chengtao Wen, Yi Zhang and her group, Runting Shi, Yi Wu, Yangbo Zhu, Hui Tan, Yifan Yanggong and Mingyan Fan. I've also enjoyed discussions and the volleyball sessions with my bay area friends.

I've had the fortune to have a great thesis committee. My committee members are senior but warm and responsive to my needs, sharp and insightful but wise enough to encourage my own thinking, passionate and experienced but careful in choosing what is necessary and relevant as their feedback. I especially liked Jaime's witty questions and comments, my casual chats with Yiming who often singlehandedly improved my understanding of the topic, and Bruce's almost reflexive but sharp intuitions that I always needed to think deeply to answer.

Carnegie Mellon University is also a great place to do research and to learn. I benefited from talks about research or teaching from such great teachers and researchers like Manuel Blum and Jaime Carbonell. I benefited a lot from interacting with fellow students by learning from their approach to research and problem solving. I also learnt many entrepreneurial skills from several faculty and fellow students. Books like the autobiography "Models of My Life" by Herbert Simon also inspire me to aim at solving scientific problems with real impact. There are so many things that I've learnt, and so much fun I've had at Carnegie Mellon that I cannot enumerate them here.

Another important source of influence to my ideas come from my interest in philosophy and the ideas of Gottlob Frege about Language Philosophy excellently re-enacted in a seminar class taught by Professor Lu Wang at Tsinghua University. The ideas and thought processes that we learnt from closely examining Frege's study of logic, truth and knowledge in general, and First Order Predicate Logic in particular, are transformative in clarifying my views of the world and scientific knowledge.

Finally, I cannot imagine how this 6 years of PhD would have been, without Jasmine. She has been very considerate and supportive throughout the process, bringing laughters into

the boring, consolations into the depressing. I'm especially grateful for her encouragement for me to pursue what I dream of instead of what would have been convenient for us. Little Gloria was born at the last stage of my PhD, in the midst of job search, multiple papers and the defense. She has brought much fun into our lives, which again would be unimaginable if without the efforts from Jasmine and our family.

My parents in law, Kai Jin and Shuhong Liu, were college graduates at a time when college graduates were much rarer than PhDs are nowadays. They not only provided useful advices on my work and our small family, but also came over abroad to help us with our daily lives when help was most needed. They also brought with them much happiness and joy.

My parents raised me with discipline, but at the same time never imposed any ideals or values on me. They encouraged me and gave me much freedom to develop my own interests and skills. As I look back, it seemed to me that every decision about my education was made without hesitation, and all for my benefit. I feel very lucky and am grateful for all their efforts.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Web search engines like Google have popularized the use of search technology to the extent that search has become part of our daily life. These Web search engines have also made search so easy to use that an ordinary search user would be satisfied with the results from the search engine most of the times. Some would even think that search is a solved problem.

This dissertation argues that search is far from being solved. As long as machines cannot perfectly understand human language as humans do, search is an unsolved problem. Some may still disagree, thinking that although it's probably true that machines still cannot perfectly understand human language, it is search that demonstrated that a natural language processing task can be tremendously successful even with very simple algorithms and easy to compute statistics (Singhal 2001). In fact, attempts to use more complex natural language processing techniques in retrieval have mostly been proved futile. Even people working intimately with retrieval technology have the impression that the research on basic retrieval algorithms is hitting a plateau. Some may even think that there is probably not much room for improvement. After all, the retrieval models of the 1990s and early 2000s (Okapi BM25 or Statistical Language Models) are still the standard baselines to compare to, and the go-to models of the modern day retrieval systems (Manning et al. 2008, Chapters 6 and 7).

We argue that search ranking, retrieval modeling to be specific, is far from being solved.

Firstly, the success of Web search engines is largely due to the vast and diverse set of high quality documents to retrieve from. Because of the diverse set of relevant documents out there, even if a search query is not well formulated, even if the retrieval algorithm cannot return most of the relevant documents in the collection, a few good ones will match the query and be ranked at the top of the rank list to satisfy the user. Just that these search engines can return satisfying results for most queries does not necessarily mean that the retrieval models used by these systems are successful, and the impression that the simple retrieval models are successful can be just an illusion created by the successful collection of a huge set of documents for the retrieval algorithm to search against.

Secondly, in cases where the document collection is small, where the set of documents relevant to the query is small, or where a high level of retrieval recall is needed, the current retrieval systems are still far from being satisfactory. For example, in perhaps all forms of desktop search and some forms of enterprise search, the document collection is much less diverse and much smaller than the Web, and the users can still be easily frustrated by the search systems. Even in Web search, for informational searches, users are still frequently frustrated by the current search engines (Feild et al. 2010). In legal discovery, the lawyers from both sides of the litigation care a lot about not missing any potentially relevant document, and usually spend lots of time on carefully creating effective search queries to improve search effectiveness.

Some may still ask, if search is not yet solved, why are the baseline retrieval models so difficult to

surpass, and where can we see any large improvements? We show in this dissertation that two central and long standing problems in retrieval, vocabulary mismatch (Furnas et al. 1987) and relevance based term weighting (Croft and Harper 1979; Greiff 1998; Metzler 2008), might be the culprit. We show that the two problems are directly related, the vocabulary mismatch problem being the more general version of the two. We show that the current retrieval models do not effectively model the vocabulary mismatch between query terms and relevant results. We show that term mismatch is a very common problem in search, and that a large potential gain is possible. We demonstrate several initial successes in addressing term mismatch in retrieval using novel prediction methods and theoretically motivated retrieval techniques. These techniques can automatically improve the retrieval system by making it mismatch-aware. Ordinary search users can also manually apply the query expansion technique studied in this work to further reduce mismatch and increase the effectiveness of their searches.

## 1.1 The Modern Information Retrieval System

There are many challenges involved in making a retrieval system successful. These challenges include acquiring lots of documents from many sources, estimating the quality of the acquired documents, extracting effective representations of the documents to facilitate search and other applications, ranking documents in response to a user request, presenting search results effectively, and all other efforts involved in tracking and analyzing user behavior and search engine performance.

The ranking problem is the central problem in a search engine, where all available information from the user, the search request and the document collection are used to determine the ranking of the result documents. To solve search ranking, researchers often start with simplified models, which are commonly referred to as retrieval models. Even though modern retrieval systems typically use a multitude of features for ranking documents, such as document quality and popularity estimates, user browsing and searching behavior, and other contextual information, the backbone for search ranking is usually still the standard probabilistic retrieval models such as Okapi BM25 (Robertson et al. 1995) or Statistical Language Models (Ponte and Croft 1998; Zhai and Lafferty 2001).

Current retrieval models typically use simple collection statistics to assess the importance of a query term and to score and rank result documents. Most of these models are based on the tf and idf statistics, where tf, short for term frequency, is the occurrence frequency of a term in a document, and idf, the inverse document frequency, is the inverse of the occurrence frequency of a term in the whole document collection. Tf measures how well a term represents a document. It only measures how important a term is for the document regardless of the query or the other query terms. Thus tf does not make any distinction among the terms in the query, and has nothing to do with term importance or term weighting. Idf measures how rarely a term occurs in the collection, and is used to assess the importance of a term during retrieval, i.e. term weighting. The general idea is to assign more importance to the rare terms which tend to be more discriminative, and to rank highly the documents that contain many of the rare terms many times. Take the query "text retrieval" for example, "retrieval" is the rarer term of the two, and a document containing "retrieval" is probably more likely to be relevant than a document containing "text". Idf is a widely used term importance measure, because it is very easy to compute given a document collection and fairly effective. Many popular retrieval models as well as term importance measures are based on this rareness idea.

## 1.2 The Limitation of Current Retrieval Models

This thesis is concerned with popular probabilistic retrieval models, their limitation and methods to overcome their limitation.

Even though retrieval is about finding the relevant documents for a query, these tf.idf models do not effectively model relevance. Tf is about how well a term represents a document regardless of the query, and idf measures term importance using the query independent notion of term rareness which has nothing to do with the query either. Neither captures the query dependent notion of how important terms are for retrieving relevant documents for that particular query.

We argue that what the current retrieval models often overlook, and what is at least as important as rareness, is a term's coverage over the documents relevant to the query, i.e. the percentage of the relevant documents that match or contain the term, which we call *term recall*.

When term recall can be different for different terms in the query, using only rareness to assess term importance can cause a serious problem in ranking, the emphasis problem. Suppose the user searches with the query "prognosis of a political third party in the US", and most of the relevant texts say instead "the future of third parties". Then, even though "prognosis" is a rare term, it should not bear as much weight for retrieval as the other terms ("political", "third" or "party") which match more of the relevant documents than "prognosis" does. In the extreme, if "prognosis" does not appear in any relevant document, the most highly ranked documents that contain "prognosis" will all be irrelevant. Thus, rareness alone cannot properly assess term importance in retrieval.

However, using simple collection statistics, current retrieval models cannot accurately assess term recall, and when the retrieval model only emphasizes the matching of rare terms, false positives can appear throughout the rank list, significantly decreasing retrieval accuracy. These false positives are irrelevant results that happen to contain these rare but low recall terms (e.g. "prognosis"), but fail to match the other important terms. For search engines requiring all query terms to appear in a result document, false positives can still appear throughout the rank list, which would be documents that contain multiple occurrences of the rare terms, e.g. "prognosis", but only a few occurrences of the other perhaps more important terms.

More generally, the example above illustrates a case of *term mismatch*, which happens when the vocabulary of the query does not match the vocabulary of relevant documents, causing query terms to have low recall.

The hazard of mismatch lies not only in the fact that term importance can be affected, tricking retrieval models into preferring the wrong query terms, but also that a large percentage of the relevant documents may be completely missed by the retrieval system. This is perhaps not a big problem for general Web search where top precision is usually guaranteed. For example, for popular queries like "Lady Gaga shocking pictures", a diverse and large set of relevant results exist which could match all kinds of variants of the query terms, and the user probably does not care if a few good ones are missing. However, for more recall-centric tasks such as desktop search and some cases of enterprise search where the search happens on small collections, or legal discovery or medical record retrieval, missing even just a few relevant results could lead to search failure and huge costs for the user.

In areas where the user cares most, mismatch can be particularly harmful. Take job search for example, the user may be looking for "information retrieval" jobs, but the job posts may say "text search" instead. Mismatch can easily cost the user a large percentage of the relevant job opportunities on the market, even when the user honestly and carefully formulates her query. In medical record retrieval, failing to find a relevant case because of vocabulary mismatch can sometimes be fatal. In these high-stake scenarios, experienced searchers would use more than one query to try to match more of the relevant results, and even interact with the search results from some initial queries to improve the search request. All of these

are ways to overcome the limitation of the retrieval models.

## 1.3 The Term Mismatch Problem in This Thesis

The vocabulary mismatch between query and document terms has been known to be important for retrieval. Various techniques based on query expansion, enriched documents, or semantic representations of texts (e.g. latent semantic indexing) have been proposed to solve the problem.

What's new here is that we approach the problem by formally defining term mismatch in retrieval, before trying to solve mismatch. The clear formal definition allows us to understand the nature of the problem theoretically and how serious it is practically. Quantitative analyses of term mismatch further allow us to understand the mechanisms that cause mismatch, as well as how mismatch may affect retrieval. All of these new understandings lead to the design of effective ways of predicting term mismatch and theoretically-motivated retrieval techniques to solve mismatch in retrieval.

Previous techniques that solve mismatch, such as query expansion, do not make any distinction among query terms, and are either applied to all query terms or to the query as a whole. A key idea utilized in this work is diagnostic intervention, to first diagnose which query terms have a mismatch problem by predicting how likely a term is to mismatch relevant documents, before trying to solve the mismatch problem. Thus, compared to existing work, this dissertation aims to fix terms that actually have a mismatch problem, instead of all of the terms, or the terms for which synonyms can be easily found.

## 1.4 Definition of Term Mismatch

In this work, we formally define *term mismatch* as follows.

**Definition 1.** For a document collection $\mathcal{C}$, and a given query $q$, the set of all relevant documents for the query is defined as

$$R := \{r \in \mathcal{C} : \text{ Relevant}(r, q)\} \tag{1.1}$$

**Definition 2.** For any term $t$, *term mismatch* is defined as the probability of $t$ not appearing in a document $d$ given that the document is relevant. The mismatch probability is denoted as $P(\bar{t}|R)$, which is just the proportion of relevant documents (documents in $R$) that do not contain term $t$:

$$P(\bar{t}|R) := \frac{|\{d \in R : \ t \notin d\}|}{|R|} \tag{1.2}$$

Here, $|R|$ is the cardinality of the set $R$, i.e. the number of relevant documents for the query $q$.

**Definition 3.** The complement of term mismatch, $P(t|R) = 1 - P(\bar{t}|R)$, is defined as *term recall*, which measures, for the set of documents that contain $t$, the recall of relevant documents.

These two probabilities are the focus of this dissertation.

Although this dissertation focuses on the analysis and modeling of term mismatch for query terms, a term $t$ does not have to appear in the query, neither does it need to be a natural language word. Broadly defined, a term can be any binary function that maps from a document in the collection to 0 or 1. Thus, a term can be a phrase, terms within a certain proximity, a disjunction of terms, a conjunction of terms, syntactically structured, a topic categorization of the collection documents, or any combinations of the above.

The $P(t|R)$ probability itself is not new. It is one of the two class-conditional probabilities in a two-class Naïve Bayes model, where the two classes are the relevant and the non-relevant documents for a query. The other class conditional probability is $P(t|\bar{R})$. $P(t|\bar{R})$ is accurately approximated by the term

occurrence in the collection $P(t|C)$, because the relevant set is usually very small, and $P(t|C)$ leads to the commonly used idf term weighting. In the retrieval model literature, this Naïve Bayes model is called the Binary Independence Model (Robertson and Spärck Jones 1976), and the $P(t|R)$ probability appeared (unnamed) as one of the two parts of the optimal term weight. This optimal term weight fully determined by $P(t|R)$ and idf is commonly referred to as the "relevance weight" or "term relevance", even though $P(t|R)$ is the only part about relevance. Thus, $P(t|R)$ plays an important role in the theory of probabilistic retrieval, and is at least as important as idf.

$P(t|R)$ is very easy to calculate once the user has examined documents – just count how often the term $t$ occurred in the relevant documents that the user liked. However, a search engine has to predict $P(t|R)$ before the user has examined documents, which is much harder. Simple statistics commonly used by retrieval models, for example, idf do not correlate well with $P(t|R)$ (Greiff 1998), and without the understanding that $P(t|R)$ measures term recall or equivalently term mismatch, it is difficult to design effective features for prediction. Without effective $P(t|R)$ estimates, prior research generally ignored it, for example setting it to a constant, or using simple methods to predict it, however these efforts were not especially successful. It has been known since the 1970s that more accurate $P(t|R)$ offers the possibility of significant improvements in ad hoc retrieval performance. Our recent measurements indicate that using the true $P(t|R)$ probabilities as query term weights for long unstructured queries produces 30%-80% improvement in Mean Average Precision (MAP) over state-of-the-art retrieval models on 6 standard test collections. A $P(t|R)$ prediction method does not need to be very accurate to show a significant improvement. In addition to term weighting, if techniques like query expansion can be applied, the potential gain from solving mismatch is even larger than 50%-300% from our experiments with high quality manual expansion queries.

## 1.5 Summary of Thesis Research

Overall, the thesis research shows that the term mismatch problem happens deep inside the retrieval models. This research shows that the term mismatch problem is a significant problem both theoretically and practically. It further investigates the underlying factors that may cause term mismatch and the mechanism that allows the mismatch problem to affect retrieval performance, and designs effective and efficient term mismatch prediction methods. Finally, this research applies the term mismatch predictions to improve retrieval in several principled and effective ways.

This work formally defines the term mismatch probability, points out the role term mismatch plays in probabilistic retrieval models, and how term mismatch affects both retrieval precision and recall through causing the emphasis and the mismatch problems in retrieval.

Given the quantitative measure of term mismatch, this work firstly investigates factors that lead to term mismatch, and uses term- and query-dependent statistics to model these factors. Secondly, using these statistics as features, it builds machine learning models that can predict, for a new query, how likely each term is to mismatch the relevant documents for the query. Finally, accurate mismatch predictions form the basis for several principled interventions that improve query effectiveness, for example, adjusted term weighting or selective query expansion. A specific contribution of our research is to improve retrieval accuracy using both unstructured ('keyword') and structured queries.

To understand the causes of mismatch, our research (Zhao and Callan 2010) shows that several factors contribute to term mismatch. These factors include the term not being central to the query, the concept represented by the term not being central or necessary for the information need, the term having many synonyms that appear in place of the original term in relevant documents, and the term being at a different level of abstraction than the relevant documents. Given this initial set of sources for mismatch, automatic

numeric features can be designed to predict the term mismatch probability.

Data analyses also show that the term mismatch probability varies query dependently, thus, query dependent features are needed for effective term mismatch prediction. But at the same time, for many term occurrences, the probability of mismatch does not change much from the mismatch probability for the same term in historic queries (Zhao and Callan 2012a). Several studies analyze the causes of the query dependent variation of term mismatch for the same term in different queries, and an alternative term mismatch prediction method is proposed utilizing the term mismatch probabilities from previous occurrences of the same term in similar query contexts. While the initial prediction method (Zhao and Callan 2010) is computationally expensive, requiring an initial retrieval and expensive computation over the top ranked documents, this new prediction method is more efficient using historic information of the query terms and efficient query dependent features to help identify previously seen query contexts that are similar to the current query.

A comprehensive retrieval evaluation is performed on 6 standard test datasets (Zhao and Callan 2010). It is possible to do so because of the wide applicability of our definition of term mismatch and our retrieval technique which are very general and only depend on the bare minimum – the relevance data that comes with these datasets.

A more interesting use of the term mismatch probabilities is to provide more controlled interventions to improve query effectiveness, for example, by diagnosing each query term and focusing query expansion on those terms that are most likely to mismatch relevant documents. The resulting queries are both effective and compact. The dissertation research (Zhao and Callan 2012b) also investigates aspects of query expansion that have been neglected in recent research. For example, it considers expansion in the context of a structured query, enabling effective selective expansion of some of the query terms.

In summary, the dissertation research revisits a classic problem in information retrieval research – the prediction of $P(t|R)$, which we view as *term recall*, or equivalently, $P(\bar{t}|R)$ which we view as *term mismatch*. Research over the years demonstrates that good estimates of these probabilities can lead to significant gains in retrieval accuracy, but the methods developed by prior research have not been sufficiently effective. This research takes an entirely new approach to the problem, and successfully demonstrates several uses of this new tool in improving retrieval.

## 1.6 Contributions

The term mismatch problem and the $P(t|R)$ prediction problem are both long standing and central problems in retrieval. This research connects the two problems, and makes progress toward solving both. Prior research did not clearly define term mismatch, and did not understand how mismatch affects retrieval. This research is the first to give a formal definition to term mismatch, as the probability $P(\bar{t}|R)$, and to apply term mismatch to improve retrieval in principled ways. Prior research provided few clues about how to predict $P(t|R)$. This dissertation understands the probability as term recall, and designs effective features for its prediction. These features allow the use of query terms from a small number of training queries to be used to learn a prediction model that can predict $P(t|R)$ for test query terms that may not exist in the training data. This new approach can be viewed as a type of transfer learning (Do and Ng 2005), because knowledge from related retrieval/classification tasks (the training queries) is used to inform the learning of new ranking models/classifiers for new tasks (test queries) that do not have explicit training information.

The significance of the research lies in the fact that we have clearly identified a problem of the core retrieval models with a great potential of improvement – the mismatch problem in retrieval. This research also provides a unique and effective tool to solve mismatch – the term mismatch probability as a way to quantify and diagnose term level mismatch. It clarifies the theoretical role of mismatch as well as its

practical significance. This new understanding that term mismatch plays a central role in retrieval theory allows us to explain the behaviors of the current retrieval models and many retrieval techniques which exist as a body of empirical knowledge in the field of information retrieval but came about largely unexplained. These new understandings about the core retrieval models and the term mismatch problem will guide the development of future retrieval techniques such as novel mismatch prediction methods, query expansion or diagnostic intervention approaches that would not have been possible without the new understandings.

Specifically, the dissertation research contributes to information retrieval by

- quantifying term mismatch in retrieval and enabling data analyses to be performed,
- pointing out the central role term mismatch plays in probabilistic retrieval theory,
- establishing a concrete connection between optimal term weighting and term mismatch,
- characterizing term mismatch, its variation and its causes through data analyses,
- designing query dependent features to accurately predict term mismatch,
- analyzing the nature and the causes for the query dependent variation of the mismatch probability for the same term across different queries,
- improving ad hoc retrieval through term weighting based on mismatch predictions,
- highlighting the importance of query expansion in the Conjunctive Normal Form for solving mismatch,
- and improving query expansion by using term mismatch as a diagnostic tool.

## 1.7 Organization of the Dissertation

The rest of the dissertation is organized as follows. Chapter 2 introduces prior research related to the term mismatch problem and its solutions. The rest of the dissertation is organized in four parts. The first part includes Chapters 3 and 4, which defines the term mismatch problem, discusses how it affects retrieval performance, and analyzes the $P(t|R)$ probability and its variation using a set of exploratory data analyses aiming to understand whether query dependent features are necessary for $P(t|R)$ prediction. The second part includes Chapters 5 and 6, which predicts the term mismatch probability $P(t|R)$ by designing features that model the possible causes of mismatch (discussed in Chapter 3), and further applies the predictions in retrieval term weighting. The third part includes Chapter 7, and addresses the efficiency problem of the 2-pass prediction method designed in Chapter 5 which requires an additional retrieval step to generate the features. A more efficient 1-pass $P(t|R)$ prediction method is designed based on the understanding about the query dependent variation of $P(t|R)$ (reported in Chapter 4). The fourth part, Chapter 8, uses query expansion to more effectively solve mismatch, and applies $P(t|R)$ predictions to develop a new type of retrieval intervention – diagnostic term expansion. Chapter 9 summarizes the work and its implications, and presents directions for future research.

# Chapter 2

# Related Work

Term mismatch and relevance based term weighting are two central and long standing problems in retrieval, and a large body of prior work is related to this dissertation research. This chapter explains how these different areas are related to the current work, and how the dissertation research is different.

Relevance based term weighting exists in various types of retrieval models (Section 2.1). In particular, probabilistic retrieval models (Section 2.2) provide the theoretical foundation for this research. These foundational theories include probabilistic retrieval theory (Section 2.2.1) as well as the Relevance Model in the language modeling framework (Section 2.2.6). In practice, for ad hoc retrieval, no relevance information is available for the test topics, and the term relevance probabilities $P(t|R)$ need to be predicted. Prior approaches that tried to predict $P(t|R)$ probabilities are reviewed in Section 2.3. More generally, term weight prediction methods are also related because of their use of predicted term weights to improve retrieval, and are reviewed in Section 2.4.

It is widely accepted that term mismatch is an important problem in retrieval. Even though there has been no clear understanding of what exact role term mismatch plays in the retrieval models and the retrieval process, a plethora of methods were proposed to solve mismatch. They worked from the document's end, the query's end, or both ends, and are reviewed in Section 2.6. Semantic analysis of texts or queries (Section 2.7) is a standard technique to improve semantic level matching in retrieval. Examples include concept identification in queries or synonym identification for query terms.

Another aspect of this research is diagnostic interventions that improve problem areas of a query. Section 2.8 discusses prior research that predicted query difficulty. Although not directly related to the mismatch problem, predicting problems that a query is suffering from is the first step toward retrieval interventions based on query level diagnosis, and is a direct generalization of how our term level mismatch diagnosis helps guide expansion and solve mismatch.

## 2.1 Relevance Term Weighting in Different Retrieval Models

The problem of relevance term weighting aims to discover a function $f(t, R)$ which assigns an importance value (the output of the function) to the term $t$ based on the relevant set of documents $R$ for the given query.

Many retrieval models use such a function to model the relevant class and to improve retrieval.

In the Vector Space Model, the relevant set of documents is modeled during Rocchio feedback (Rocchio 1971) using all the judged relevant and non-relevant documents. The average vector of all the relevant documents provides such a model of the relevant class. Result documents can be scored according to how similar they are to the relevance vector.

In probabilistic retrieval models (Robertson and Spärck Jones 1976), the $P(t|R)$ probabilities specify a model of the relevant class – the function $f(t, R)$.

In statistical language modeling (Ponte and Croft 1998; Zhai and Lafferty 2001), $f(t, R)$ is instantiated as the multinomial distribution $P_m(t|R)$ in the Relevance Model (Lavrenko and Croft 2001), which is also typically estimated using (pseudo) relevance feedback. The KL-divergence from a result document to the relevance model is used to score and rank the document.

Sometimes this query dependent value $f(t, R)$ for each query term can also be applied as a user term weight to modify the term importance of each query term. For example, under the language modeling formalism, applying $f(t, R)$ as in the Relevance Model is equivalent to applying the value as a user term weight. In other cases like Okapi BM25, the relevance based term weight is separate from the user term weight which is usually instantiated as the query term frequency.

The section below explains how $P(t|R)$ stems from the basic probabilistic retrieval models as a model of the relevant class, as well as details about how different probabilistic retrieval models could allow query dependent term weighting.

## 2.2 Probabilistic Retrieval Models

Probabilistic retrieval models are one particular formalism of retrieval models that contain a model of the relevant class. In probabilistic retrieval models, the relevant class is modeled by the $P(t|R)$ probability, the subject matter of this thesis. For a given query, $P(t|R)$ is the probability of observing term $t$ in a document given that the document is relevant to the query. It appeared in probabilistic retrieval models such as the Binary Independence Model early on, but none of the previous work named the probability, or demonstrated an understanding of this probability as being term recall or related to term mismatch.

### 2.2.1 Roots in the Binary Independence Model

The Binary Independence Model (BIM) is a basic probabilistic retrieval model. Later models such as Okapi BM25 are built on top of the BIM. In this section, we review the BIM, in particular how it models the relevant class, and what theoretical role the model of the relevant class plays in probabilistic models.

The Probability Ranking Principle states that the best retrieval effectiveness is achieved when documents are ranked in decreasing probability of relevance to the user that submitted the request, where probabilities are estimated using all available evidence (Robertson and Spärck Jones 1976). Further assuming binary term occurrences and that term occurrences are conditionally independent of each other given the relevance class, the Probability Ranking Principle leads to the following optimal retrieval model, which is usually instantiated as ranking by the Odds ratio,

$$\frac{P(R|d)}{P(\bar{R}|d)} \propto \prod_{q_i \in q \cap d} \left( \frac{P(q_i|R)}{1 - P(q_i|R)} \cdot \frac{1 - P(q_i|\bar{R})}{P(q_i|\bar{R})} \right) \tag{2.1}$$

where $R$ is the set of relevant documents, $\bar{R}$ is the set of non-relevant documents, $d$ is a document to be ranked, $q$ is the query, and $q_i$ is a query term in $q$. This is known as the Binary Independence Model (BIM) (Robertson and Spärck Jones 1976). Broken down to each query term, the BIM yields the well known Robertson-Spärck Jones (RSJ) term weight for that term.

$$RSJ(t, q) = \frac{P(t|R)}{1 - P(t|R)} \cdot \frac{1 - P(t|\bar{R})}{P(t|\bar{R})} \tag{2.2}$$

Two probabilities determine the final term relevance score: $P(t|R)$ and $P(t|\bar{R})$. The BIM is just a binary Naive Bayes model in the machine learning literature, where these two probabilities for a term are just the class conditionals – the only parameters for the Naive Bayes model.

### 2.2.2 Estimating the Model Probabilities

$P(t|\bar{R})$ is closely associated with the document frequency of the term divided by the collection size ($P(t|C)$), as demonstrated in the predictive version of BIM (Robertson and Spärck Jones 1976) and also confirmed by the data analysis performed by Greiff (1998); it leads to the well-known inverse document frequency (idf) metric. $P(t|R)$ is term recall: The probability that a document relevant to the given query contains t. $P(t|R)$ can be accurately estimated given relevant documents for the query, which has been called the retrospective case by Robertson and Spärck Jones (1976). In the retrospective case, $P(t|R)$ is just the proportion of relevant documents that contain the term t. Greiff (1998) showed that $P(t|R)$ is not very well correlated with idf. Historically $P(t|R)$ has been more difficult to predict than $P(t|\bar{R})$, and people either chose to ignore $P(t|R)$ in their models or used simple methods to tune it.

Overall, $P(t|\bar{R})$ measures a term's recall of irrelevant documents, while $P(t|R)$ measures the *term recall* of relevant documents.

$P(t|\bar{R})$ is both easy to predict and well understood, while $P(t|R)$ (term recall) or $P(\bar{t}|R)$ (term mismatch) are the focus of the dissertation research.

### 2.2.3 Terminology & Understanding $P(t|R)$ from Different Viewpoints

Prior work has a limited understanding of $P(t|R)$. The RSJ term weight (Robertson and Spärck Jones 1976) – the product term in Equation 2.1 – has been called "term relevance" by Salton and "relevance weight" by Spärck Jones (van Rijsbergen 1979), but not the $P(t|R)$ probability itself (which is effectively the only part of the weight about relevance). For all the terms collectively, this probability distribution is treated as a model of the relevant class, e.g. the Lavrenko Relevance Model (Lavrenko and Croft 2001).

This thesis research understands $P(t|R)$ and $P(\bar{t}|R)$, two closely related probabilities, from different viewpoints. These new viewpoints allow us to more accurately capture what these probabilities are, to more effectively reason about what may affect them, and to suggest new ways of how they can be used in practice.

First, $P(t|R)$ measures the percentage of the relevant documents that can be returned by the term $t$, thus, measures *term recall*. $P(\bar{t}|R)$, being the complement of term recall, measures *term mismatch*.

Generally the probability $P(a|b)$ measures how likely event $a$ occurring is a necessary condition for $b$ (Goertz and Starr 2002, pg. 10) (or whether event $a$ must occur in order for $b$ to be true). $P(t|R) = 1$ means term $t$ must appear in the document in order for it to be relevant, while, low $P(t|R)$ means $t$ is unlikely necessary for relevance. Thus, $P(t|R)$ measures a term's *necessity* for relevance, or simply *term necessity*.[1]

We make two distinctions to further clarify the definition of term necessity or recall. Firstly, term necessity or term recall is not concept necessity. The concept represented by the term can be necessary for relevance, while the term may be less necessary, for example, when synonyms of the query term appear in relevant documents, causing mismatch of the query term. Secondly, term necessity or recall is the necessity to relevance, not the necessity to good performance or anything else. For example, some of the stopwords (e.g. "the", "of", "is"), because of their general prevalence in the collection, also occur

[1]Note, necessity here is an association measure for two random variables, and does not necessarily imply a logical necessity relation, unless in the extreme case where the probability is 1 in which case the relation is almost always a necessity relation.

frequently in the relevant set. Thus, they are necessary to relevance, but they do not need to be in the query to achieve decent retrieval performance.

Second, the new understanding allows us to reason about what may affect them. For example, the understanding of necessity urges one to reason why a user specified query term may become unnecessary for relevance. It is important to realize that low necessity does not imply that a term is irrelevant to the information need. Synonyms appearing in place of the query term in relevant documents can cause term mismatch, lowering necessity. Understanding what factors may affect term necessity or mismatch can help in designing features to predict it for unseen terms in new queries.

Third, the new understanding suggests new ways of using the probabilities in practice, for example, in helping query reduction or expansion. Intuitively, terms unnecessary for relevance can be safely removed or replaced, while low necessity terms should be expanded with proper expansion terms, to increase overall recall after expansion.

Since this thesis is about term recall, it is also important to clarify another closely related probability, *term precision* or *term sufficiency*[2]. The *term precision* probability, defined as $P(R|t)$, measures the precision of the result set when only using term $t$ for retrieval. It is directly determined by term recall and the document frequency of term $t$, $\text{df}(t)$:

$$\begin{aligned} P(R|t) &= P(R|t,C) \\ &= \frac{P(t|R)\cdot P(R|C)}{P(t|C)} \\ &\propto \frac{P(t|R)}{P(t|C)} && (P(R|C) \text{ is a constant for a given query}) \\ &\propto \frac{P(t|R)}{\text{df}(t)} && (|C|, \text{the size of the collection, is a constant}) \end{aligned} \tag{2.3}$$

This is directly derived from the Bayes' formula, assuming that any document comes from the collection $C$.

From the derivation we can see that rare collection terms can be highly precise only when they have a high recall rate. This also means if we assume $P(t|R)$ to be a constant, as prior research typically did, then term precision is roughly estimated from the inverse of the document frequency of the term.

This dissertation focuses on understanding and predicting the probability $P(t|R)$. The rest of this document refers to $P(t|R)$ as term recall, and its complement $P(\bar{t}|R)$ as term mismatch.

### 2.2.4 2-Poisson Model

The 2-Poisson model is an indexing model which aims to answer the question how well a term represents a document, which is different from the retrieval problem where the question is instead how important a term is for returning relevant documents for a query. A distinction of two classes of documents – elite versus non-elite – is made. The two-Poisson model assumes that for a given term, the elite documents are those that are more likely to be about the topic described by the term, while the non-elite documents are less likely to be about the term. The Binary Independence Model is different, and assumes a classification of relevant versus non-relevant documents that is dependent on a particular query.

The difference between representation and relevance is an important one that represents an important development in library science. Traditionally, library science was very concerned with the representation problem of selecting indexing terms that best represent a document, to facilitate keyword indexing or cataloging, while modern information retrieval research is directly aimed at the relevance problem.

The distinction between representation and relevance also dictates how the respective models estimates their parameters. The two Poisson model can be estimated offline with only access to the collection, and a

[2]This term precision is different from the term weight function referred to as "term precision" in (Yu et al. 1982).

total of four parameters need to be estimated for each vocabulary term. The Binary Independence Model parameters are query dependent, with two parameters for each term for a given query.

The ideas of representation and relevance are orthogonal to each other, and can be built into a single more advanced retrieval model. For example, the Okapi BM25 model described in the section below uses both ideas to incorporate term occurrence frequency as well as term relevance information into one retrieval model.

This dissertation is concerned with the relevance problem, but the representation problem is also studied by understanding how term frequency and document length information impacts the modeling of term occurrences in the relevant class of documents and how it affects retrieval performance. Section 6.1.2.2 provides experiments and discussions about the representation problem in this dissertation.

### 2.2.5 Okapi BM25

The later and more advanced retrieval models such as the Okapi BM25 model (Robertson et al. 1995) build on top of the Binary Independence Model to include document length information and term frequency information from both the query and the documents. The basic idea is to gradually decrease the relevance score of a document assessed by the Binary Independence Model as the occurrence frequency of a query term decreases in a result document. When a document contains a query term very frequently, i.e. the document is about the term under the two-Poisson assumptions, the final relevance score of the document is directly calculated from the relevance based term weights from the Binary Independence Model. When the occurrence likelihood of a query term in a document becomes extremely small, the final relevance score for the document contributed by that term diminishes to 0.

The ideal BM25 model is defined as follows,

$$\begin{aligned}\text{BM25}_{\text{Ideal}}(D,q) = \quad & \sum_{t\in q\cap D} \log\left(\frac{P(t|R)}{1-P(t|R)}\cdot\frac{1-P(t|\bar{R})}{P(t|\bar{R})}\right)\cdot\frac{\text{C}(t,D)\cdot(k_1+1)}{\text{C}(t,D)+k_1(1-b+b\frac{|D|}{avgdl})}\cdot\frac{qtf\cdot(k_3+1)}{k_3+qtf} \\ & +k_2\cdot ql\cdot\frac{avgdl-|D|}{avgdl+|D|}\end{aligned} \tag{2.4}$$

where $P(t|R)$ is the term recall of $t$ for query $q$. $P(t|\bar{R})$ is approximated with $P(t|C)=\frac{df_t}{N}$, $N$ being the number of documents in the collection. This first part of the BM25 model comes directly from the Binary Independence Model, and deals with the relevance problem. The rest of the model deals with the representation problem. $\text{C}(t,D)$ is the term frequency for $t$ within document $D$, and $qtf$ the term frequency in the query. $|D|$ is the length of the document measured in the number of tokens in the document, $avgdl$ is the collection average document length, and $ql$ is the length of the query in number of words. The free parameters are $k_1$, $k_2$, $k_3$ and $b$.

Breaking the formula down, the BM25 model uses the RSJ weights to assess term importance for retrieval and uses term frequency $\text{C}(t,D)$ to measure how well term $t$ represents the document $D$. When $\text{C}(t,D)$ becomes large, the BM25 model would directly use the RSJ weight of a term as its contribution to the final ranking score of a document (Robertson and Walker 1994).

In practice, $P(t|R)$ is set at 0.5 assuming no known relevant documents, and the final form of BM25 becomes

$$\begin{aligned}\text{BM25}_{\text{Practice}}(D,q) = \quad & \sum_{t\in q\cap D} \log\left(\frac{N-df_t+0.5}{df_t+0.5}\right)\cdot\frac{\text{C}(t,D)\cdot(k_1+1)}{\text{C}(t,D)+k_1(1-b+b\frac{|D|}{avgdl})}\cdot\frac{qtf\cdot(k_3+1)}{k_3+qtf} \\ & +k_2\cdot ql\cdot\frac{avgdl-|D|}{avgdl+|D|}\end{aligned} \tag{2.5}$$

Further assuming $k_2$ to be 0 and $k_3$ to be infinity leads to the widely used BM2500 formula (Gao et al.

2002),

$$\text{BM2500}(D,q) = \sum_{t \in q \cap D} \log\left(\frac{N - df_t + 0.5}{df_t + 0.5}\right) \cdot \frac{\text{C}(t,D) \cdot (k_1+1)}{\text{C}(t,D) + k_1(1 - b + b\frac{|D|}{avgdl})} \cdot qtf \tag{2.6}$$

Because of the intimate connection with BIM, it is straightforward to insert the term recall weights back into the BM25 or BM2500 models as done in Equation 2.4.

### 2.2.6 The Relevance Model – Unsupervised Prediction of Relevance

The basic language model retrieval models score a document by its probability of generating the query, thus, do not contain a relevance variable, and do not model relevance in any way, as pointed out by Spärck-Jones et al. (2003)[3]. Lavrenko and Croft (2001) proposed to estimate an ideal multinomial term distribution called the relevance model ($P_m(t|R)$) for each query, and to score a document model according to its KL-divergence with the relevance model. Their model represents a successful attempt to address the relevance problem in the language modeling framework, and to estimate the relevance probability in an effective and query dependent way.

The final retrieval score given by the relevance model is,

$$Score(R,D) = KL(R\|D) = \sum_{t \in R} P_m(t|R) \cdot \log\left(\frac{P_m(t|R)}{P(t|D)}\right) \tag{2.7}$$

If the user specifies an accurate description of the information need in the initial query by repeating more important terms a larger number of times in the query, then the generation probability for a document $D$ to generate the query $q$ becomes

$$Score(q,D) = \log P(q|D) = \sum_{t \in R} \text{qtf}(t,q) \cdot \log P(t|D) \tag{2.8}$$

where $\text{qtf}(t,q)$ is the query term frequency – the number of times that $t$ occurs in the query $q$.

Comparing Equation 2.8 with Equation 2.7, $\text{qtf}(t,q)$ shares the same role as the term relevance probability in the Relevance Model. This means the KL-divergence based Relevance Model is a natural generalization of the generative language model by relaxing the assumption that the user specifies her query term weights accurately. This way of weighting terms is equivalent to weighting query terms using the $\#weight$ operator in the Indri query language (Strohman et al. 2005), which was originally designed to allow users to specify weights for the query terms. Thus, we call these term weights *user term weights*, when they are applied to the language model retrieval model as shown in Equation 2.7.

#### 2.2.6.1 Relevance Model Estimation

Lavrenko and Croft (2001) proposed to directly estimate $P_m(t|R)$ in an unsupervised way by decomposing the probability along the top N returned documents from an initial retrieval (pseudo relevant documents).

$$P_m(t|R) = \sum_{D \in TopN} P_m(t|D) \cdot P(D|R) \tag{2.9}$$

[3]The fact that basic language models do not model relevance should not be surprising, as the predictive case of the Binary Independence Model doesn't contain anything about the relevance variable either, even though the derivation of the model starts from relevance.

This step of estimation only makes one conditional independence assumption, that $P_m(t|D,R) = P_m(t|D)$. This assumption says that how well a term represents the document should not be affected by the fact that the document is relevant to the current query. This is a fairly reasonable assumption because a document that is about "Obama" is still as much about "Obama" no matter whether that document is relevant to a "politics" query, on a "Obama family tree" query or any other query.

Prior pseudo relevance feedback (PRF) methods like Rocchio feedback simply assume that a set of top N ranked documents are relevant to the query.

An important contribution of the Relevance Model work is that it weakens the above pseudo relevance assumption by bringing in a probabilistic notion of document relevance – $P(D|R)$. This means in the Relevance Model, not all top ranked documents are treated the same, and the final term relevance probability $P_m(t|R)$ needs to take into account the likelihood that each individual pseudo relevant document is relevant to the query.

One way that Lavrenko and Croft (2001) proposed to estimate $P(D|R)$ is to assume that $P(D|R)$ can be estimated using the generation probability for the document $D$ to generate the original query $q$, i.e.

$$P(D|R) = P(q|D) \tag{2.10}$$

This is a fairly strong assumption, perhaps even a very inaccurate one, about the relevance probability of the top returned documents from the initial retrieval, but at the same time, perhaps there are no other easy and more accurate ways to estimate this probability.

Together, Equations 2.7 2.9 and 2.10 fully specify the model.

#### 2.2.6.2 Relation to the Dissertation Research

The Relevance Model and the current thesis research are closely related in their goal of estimating term relevance distributions, but different in their approaches and intended scopes. The paragraphs below provide a detailed account of the similarities and differences.

Both the Relevance Model and the term recall probability try to estimate a relevance based term distribution, or a query dependent notion of term importance/relevance. The Relevance Model (Equation 2.7) provides the representational power to insert any distribution as the relevance distribution for a given query, and provides the mechanism to incorporate the term recall probabilities into the language model framework. Generally, the Relevance Model provides a theoretically justified means of using relevance based term weighting in the language model framework. The original work also provided several different ways of estimating the relevance term distribution, but these were not intended to be the only ways (nor the best ways) to estimate the relevance model.

In this sense, the thesis research can be seen as a better way of estimating the relevance model, thus, providing better term weights, and ultimately, can be directly applied to the language model retrieval models by plugging in a relevance term distribution estimated from the supervised term recall predictions.

In addition, the decomposition and estimation assumptions made in the Relevance Model can also be applied to estimate term recall probabilities. For example, when estimating $P_m(t|R)$, Lavrenko and Croft (2001) set out to compute the relevance term distribution $P_m(t|R)$ directly, by decomposing the probability along top ranked documents from an initial retrieval. This provides a way to estimate the probability $P_m(t|R)$ from term frequency in top ranked documents $P_m(t|D)$, and document relevance scores $P(D|R)$. This estimation technique can be used in term recall prediction as well, but the thesis research focuses on supervised estimation methods. For example, the Relevance Model assigns different document relevance scores ($P(R|D)$) to different top ranked documents, outperforming traditional pseudo relevance feedback methods. This more advanced treatment can be used to estimate term recall as well.

The most important difference between the Relevance Model research and this dissertation lies perhaps in the scope of problems they are trying to solve. The Relevance Model research is mainly concerned with the relevance problem, using a multinomial distribution over vocabulary terms to model the relevance distribution, and can be applied to a variety of retrieval tasks like image retrieval and cross-lingual retrieval. The dissertation research focuses on text retrieval and on the term recall probability, thus connects the relevance problem to the more general vocabulary mismatch problem.

There are several detailed differences between the Relevance Model estimation of $P_m(t|R)$ and the term recall $P(t|R)$ predictions in this dissertation. Firstly, the Relevance Model uses unsupervised estimation, while this dissertation uses supervised learning to predict the term relevance distribution. Secondly, the relevance term distribution in the Relevance Model – $P_m(t|R)$ – is a multinomial distribution over all terms, instead of a Bernoulli distribution over term occurrence or non-occurrence as in the term recall probability $P(t|R)$. This affects how the probabilities are estimated, but does not affect generalizing the estimation techniques of the Relevance Model into predicting term recall. For example, the multiple Bernoulli distributions used in term recall estimates can always be normalized into a multinomial distribution, thus, recall probabilities can be plugged into the Relevance Model as term weights with a simple normalization step. The same decomposition over top documents can also be used in the Bernoulli term recall estimations, for example by using the Bernoulli term occurrence $P(t|D)$ instead of the multinomial $P_m(t|D)$. Thirdly, the two methods differ in how they apply the term relevance distributions in retrieval. The Relevance Model uses the term relevance probabilities to weight expansion terms in a bag of word form, and this weighted expansion query is further combined with the original unweighted query for final retrieval. Our experiments show that this bag of word style expansion and combination with original query is appropriate for short and accurate queries, but perhaps not necessary for long queries. The dissertation research focuses on more principled interventions such as query term reweighting and Conjunctive Normal Form expansion which addresses both the relevance and the mismatch problems. Performance differences between the two methods are discussed in the term-weighting experiments in Chapter 6 and expansion experiments in Chapter 8.

## 2.3 Prior Work on $P(t|R)$ Prediction

Ever since the Binary Independence Model, researchers have known that accurate estimates of $P(t|R)$ would result in large gains in retrieval. However, there have been only three efforts made to try to estimate the $P(t|R)$ probability to improve retrieval performance. These techniques can be categorized into unsupervised and supervised cases. The dissertation research is the first to use query dependent features to predict $P(t|R)$ in a supervised learning framework, which achieved state-of-the-art prediction accuracy.

### 2.3.1 The Croft/Harper Combination Match

Croft and Harper (1979) were perhaps the first to tune $P(t|R)$ in ad hoc retrieval. They treated $P(t|R)$ as a tuned constant (the Croft/Harper Combination Match), independent of the term or the query.

$$\begin{aligned}
\log \frac{P(R|d)}{P(\bar{R}|d)} &= \sum_{q_i \in q \cap d} \left( \log \frac{P(q_i|R)}{1-P(q_i|R)} + \log \frac{1-P(q_i|\bar{R})}{P(q_i|\bar{R})} \right) + c \qquad (c \text{ is a constant}) \\
&= \sum_{q_i \in q \cap d} \left( \log \frac{P}{1-P} + \log \frac{1-P(q_i|\bar{R})}{P(q_i|\bar{R})} \right) + c \qquad (P \text{ is a tuned constant}) \quad (2.11) \\
&= \sum_{q_i \in q \cap d} \left( \log \frac{1-P(q_i|\bar{R})}{P(q_i|\bar{R})} \right) + |q \cap d| \cdot \log \frac{P}{1-P} + c
\end{aligned}$$

where $P$ is the tuned constant, and $|q \cap d|$ is the number of unique query terms contained in the document $d$.

This approach continued into the early 1990s in the INQUERY retrieval system (Turtle and Croft 1990; Callan et al. 1992). The original work tuned $P(t|R)$ and reported the best performance. 0.9 was found to be most effective, which the authors suspected to be due to the manually indexed queries for the Cranfield collection (C1400I), where indexers may tend to index words that may appear more frequently in relevant documents. The work also suggested tuning $P(t|R)$ for different collections. It was set at 0.4 in the INQUERY system, and later tuned with a heuristic formula of df and the average document length of the collection (Allan et al. 1996), at which point, the estimation becomes term dependent. Although, their tuning was not in, strictly speaking, a supervised learning setting, this is still one of the first attempts to adapt this probability, and to show that it does improve retrieval performance.

When applied to BIM, treating $P(t|R)$ as a constant is equivalent to adding an extra feature – the number of matching query terms for a document – into the ranking model, besides the traditional idf based ranking scores. This feature is nowadays commonly used in learning to rank scenarios (Liu et al. 2007). It is quite clear from the combination match model expressed in Equation 2.11 that when $P$ is set at a value larger than 0.5, the Combination Match model would reward documents that match more of the distinct query terms. When smaller than 0.5, the model would penalize documents that match more of the unique query terms. How far $P$ is away from 0.5 determines the weight of this number-of-matching-query-terms feature, i.e. how much to reward or penalize.

In addition, Croft and Harper (1979) also proposed to estimate the probability from top retrieved documents from an initial run, and showed performance gain over the baseline methods. This is probably the first pseudo relevance feedback work. Different from later PRF methods, the pseudo-relevant documents were only used to estimate term weights for the query terms (which is similar to this thesis) or expansion terms. Expansion terms were extracted from the whole corpus, not just the pseudo-relevant documents.

### 2.3.2 Idf Based Predictions

Two other pieces of research used idf to predict $P(t|R)$ in a supervised way.

Greiff (1998) examined the two model probabilities of the Binary Independence Model using exploratory data analyses. He showed that $P(t|\bar{R})$ can be very accurately approximated by the occurrence probability of the term $t$ in the whole document collection $P(t|C)$ leading to the commonly known idf formulation, and that $P(t|R)$ is much more difficult to estimate. Further data analyses showed that a slight correlation exists between idf and $P(t|R)$. Guided by the exploratory data analyses, Greiff (1998) used a 3-piece linear function of idf to predict the overall term weighting, which includes both $P(t|R)$ and $P(t|\bar{R})$. Even though the correlation between $P(t|R)$ and idf is not particularly strong, experiments still showed some statistically significant improvement over the Binary Independence Model baseline.

More recently Metzler (2008) predicted $P(t|R)$ as a linear function of document frequency (df). The work generalizes the idf based term weighting, and was called generalized idf. But again, because df was the only feature used for prediction, the performance gain was only shown over the more simplistic BIM baseline.

Croft and his colleagues are among the few to make serious attempts to estimate $P(t|R)$, but their primary evidence for estimating it was document frequency or idf. Success was limited, even though prior work (Robertson and Spärck Jones 1976; Greiff 1998) found much larger (80% - 100%) *potential* gains over the BIM.

## 2.4 Prior Work on Term Weight Prediction

One standard way to apply the $P(t|R)$ predictions is to use them to weight query terms. Thus, a broad body of research that predicts term weights is related. There has been a renewed interest in learning term weights in the recent years, especially with the long query initiative [4] (Bendersky and Croft 2008; Kumaran and Carvalho 2009). Given the common goal of predicting term weighting to improve retrieval, this large body of term weight prediction research likely shares similar effective features or prediction methods with the dissertation research on $P(t|R)$ prediction. Some of these term weight prediction methods are reviewed in this section.

What's unique to the dissertation research is the understanding of $P(t|R)$ as term recall, which provides the possibility to interpret and understand why some features work while others don't. Because of the lack of this interpretability, prior term weight prediction methods cannot provide any guidance for the design of effective features, except trial and error.

### 2.4.1 Berkeley Regression

Earlier work includes the Binary Independence Indexing model (Fuhr and Buckley 1991) and the Berkeley Regression (Cooper et al. 1992).

Fuhr and Buckley (1991) learned to predict indexing weights from relevance information to provide a better probabilistic index. The target is to predict how well a term represents a document that the term appears in. Simple collection statistics are used as prediction features, for example, tf, idf, maximum number of times a term appears in the document (maximum tf), document length, etc.. The model learns optimal combination weights for these features or polynomials of these simple features based on hundreds of relevance judgments as training data.

These indexing weights of terms do not depend on any specific query, but only the term and a document. They serve a very different role as the query term weights in this dissertation. The only similarity is in the use of relevance information as training data and the use of statistical regression to obtain the final prediction formula.

Berkeley Regression (Cooper et al. 1992) learned a whole retrieval model from relevance judgments, where the logistic regression model was built on top of standard features used by traditional retrieval models such as tf and idf. Because the regression weights on the features and the features themselves are query independent, the same term in different queries will receive the same predicted weight. Because of this restricted prediction formalism, only marginal improvements is shown over tf*idf baselines (Gey 1994). This dissertation predicts only the $P(t|R)$ part of the retrieval model instead of a full retrieval model, and designs effective query dependent features.

### 2.4.2 Learning to Rank

The modern Learning to Rank research (Liu et al. 2007) aims to learn an optimal ranking for a list of objects. It typically uses features based on query-document pairs, and learns prediction models that combines these input features to form an output ranking model. Learning to Rank methods typically use queries with relevance judgments to generate training data, and aims to learn optimal ranking models. In this sense, the Berkeley Regression can be seen as a special Learning to Rank model.

Learning to Rank aims to directly optimize ranking effectiveness. Measures such as Mean Average Precision, object-pairwise ranking consistency and list-wise consistency (Cao et al. 2007) are frequently used as the optimization criteria for learning the ranking model. Learning to Rank methods typically use

[4] $http://videolectures.net/cikm08\_croft\_upis/$

basic retrieval models such as Okapi BM25 or language models as input features. These features are query dependent, but do not offer much more information than the basic retrieval models do. The final output model is a complex retrieval model that uses the basic retrieval models as components.

Because of the general setup of the Learning to Rank framework, new features (e.g. features based on the query-term-document triple) can be easily included into the framework. However, the framework itself does not provide any guidance as for what kinds of features might be effective.

### 2.4.3 Regression Rank

More recently, Lease et al. (2009) used regression with features such as tf, idf, tf in Google n-gram data etc. to predict term weights of query terms. They directly optimized retrieval performance (MAP) by tuning the user term weights of a baseline retrieval model. Compared to earlier work in Learning to Rank (Liu et al. 2007), the key novelty lies in the ability to fine tune term weights query dependently, while learning to rank typically only operates on coarse level features based on the query-document pair.

The main departure from previous work on supervised term weight prediction is to predict term weights query dependently. Regression Rank used several new and query dependent features to predict optimal term weights for query terms, including the part-of-speech of a query term, a term's location in the query and information of the surrounding terms in the query.

Comparing to the dissertation research, Regression Rank tuned user term weights to directly optimize retrieval performance (MAP), without a relevance probability prediction task in the middle. Thus its formulation is more direct than term recall prediction, but also harder to solve. To search for the optimal term weights for a training query, intensive computation is required. This is because the objective function (MAP) is not continuous, requiring fairly exhaustive search through the parameter space which is exponential over the number of terms in the query. Even with exhaustive search, because MAP is a non smooth objective function, the learned weights could easily overfit on the query, rendering the weights not generalizable to other queries. Thus, Lease et al. (2009) used additional sampling to stabilize the weights. Furthermore, because there is an infinite number of optimal term weight combinations for a query, each being a constant factor off the others, Regression Rank has to normalize term weights within each query to make the problem tractable, which produces weights that cannot be compared easily across different queries. This is not generally viewed as a serious flaw, however the current dissertation research suggests that term mismatch probabilities estimated for one query are often effective for the same terms in other queries. Pilot study in the dissertation research also shows that oracle Regression Rank performance is only 10-15% better than the oracle recall-based term weighting in retrieval, thus the main contribution of Regression Rank may be due to indirectly estimating term recall.

Because Regression Rank used heavy sampling and searching procedures to estimate the gold standard term weights, the gold standard weights are hard to come by. The search space increases exponentially as query length increases. This becomes intractable for very long queries. Bendersky et al. (2010) used a small set of term features to predict term weights, but instead of the search for the best term weights, they tuned the feature weights to optimize retrieval performance (Mean Average Precision - MAP) directly in an efficient manner.

The generality of the regression models, and more generally Learning to Rank, allows the incorporation of any kind of features, no matter based on query-document pairs or query-term-document triples. But at the same time, a weakness common to all these approaches is that no guidance is provided as to what kinds of features might be effective and should be used to predict optimal term weights.

### 2.4.4 Key Concept Detection

Bendersky and Croft (2008) observed that current retrieval models typically fail on long queries because the top ranked results would emphasize certain aspects/concepts in the query, but miss certain key concepts in the query. They proposed to group query terms into so called concepts, then to assess the keyness of the concepts in the query, and to weight the concepts according to their predicted keyness. Similar to Regression Rank, concept tf, idf, and occurrence frequency in Google n-grams were used as features for concept keyness prediction. In addition, two new features were used, which measured the occurrence frequency of the concept in a query log as part of a query, and the occurrence frequency in the query log as an exact query. Bendersky and Croft (2008) used a boosted decision tree model to identify key concepts in test queries. During retrieval, to increase the emphasis on these identified key concepts, the key concepts were weighted by their likelihood of being a key concept, and combined with the original query.

This work is related to the dissertation research in two major ways. First, concept keyness has been used to predict weights for concepts by Bendersky and Croft (2008), which shares some similarity with recall or necessity in the dissertation. Bendersky and Croft (2008) motivated the work by observing that many search queries fail because the top results miss some "key concepts [...] that must be [...] in a retrieved document in order for it to be relevant". This is just the definition of concept necessity. Secondly, both lines of research weight the query components to improve retrieval.

The key concept research is also different from the dissertation research in several ways. First, the dissertation research only predicts term level recall or necessity, while the key concept research first groups the query terms into concepts, and then predicts how important the term groups (concepts) are for the query. Understanding the query through identifying the concepts first seems to be a natural way to improve how the current retrieval models handle a query. In cases where precision is needed, grouping terms into concepts or even phrases can restrict the retrieval matching of the concepts and improve precision. This dissertation, on the other hand, focuses on simple term level interventions, thus does not need to handle the problem of defining and identifying concepts from queries. Second, when carrying out the experiments, Bendersky and Croft (2008) chose to do additional manual labeling to identify for each training query one key concept, while this thesis uses recall or necessity to weight query terms. Term recall by definition is just the percentage of relevant documents that contain the term, thus, the relevance judgments of a user query alone are enough to estimate it; no extra labeling is required.

Looking closely, Bendersky and Croft (2008) viewed a concept as a probabilistic-AND combination of some of the original query terms, a bag of words conjoined together, producing one single belief or relevance probability for each resultant document. There are several theoretical and practical difficulties in doing so. Firstly, the definition and modeling of a concept is itself a hard problem. It is not clear why concepts should be modeled as suggested. Why not use Boolean AND or even phrases instead of probabilistic AND? What about other ways of expressing the same concept using terms that are not present in the query? What about the different levels of granularity that a single concept can exhibit in one query? Secondly, concept keyness seemed to be a dichotomous or binary variable during the manual labeling step (Bendersky and Croft 2008), which suggests that a concept would be a logical and objective entity. But during retrieval, a probabilistic AND of terms was treated as a concept and weighted. A logical gap exists between the two different treatments of a concept. Thirdly, one would imagine the weight on a combinations of terms to be more dependent on the terms than the concept that the terms denote. Thus, it is unclear how to model the keyness or necessity for such a probabilistic AND of words, nor whether it makes sense to apply the likelihood of concept keyness as a weight for the probabilistic AND of words. Fourthly, when doing manual labeling of key concepts, Bendersky and Croft (2008) assumed only one key concept per query. However, if keyness is really the necessity of a concept, there could be multiple necessary concepts for a query. Maybe what is pursued by the key concept research is the worst matched

necessary concept in a query. Finally, the theory of combining evidence from all possible concepts to get a final relevance score for a document is not clearly defined. Bendersky and Croft (2008, Formula (3)) suggests to use a weighted sum of the concept generation probabilities, which corresponds to the $\#wsum$ operator in Indri, but in the experiments the $\#weight$ operator was used, which does a weighted sum of log probabilities.

Overall, the idea of estimating concept keyness or concept necessity is attractive. A query concept being unnecessary is in fact one of the reasons that makes a query term become unnecessary for relevance, thus, it will be important in mismatch-based query diagnosis and retrieval intervention (e.g. in the form of query reformulation). Manual annotation of necessary concepts seems to be the right thing to do for training classifiers and predicting concept necessity.

The main point here is that we should distinguish the thing we are looking for from the way we use to refer to it. For the key concept work (Bendersky and Croft 2008), "Spanish civil war" is one single concept referring to a particular event that started in 1936, and there is no arguing against this. However, the particular terms that are being used to describe the concept are very different from the concept itself. For example, even when the concept is necessary for the information need, the term "Spanish" may have a low recall probability because documents written by Spanish people would probably not mention "Spanish", but instead "1936 civil war", or "1936 coup". Thus, a better way of matching this concept is to use a query like *(Spanish OR 1936) AND ("civil war" OR coup)*. Here, we are treating the original terms almost independently, expanding each with its synonyms, and overall it will match text references to the concept more accurately than the original phrase.

## 2.5 Transfer Learning

All of these learning and prediction approaches that predict $P(t|R)$ or optimal term weights require a more general view of the retrieval modeling problem, where a retrieval model is seen as a meta-classifier responsible for many classification tasks, with each query being a classification task. The learning of the retrieval model allows information from related training tasks (training queries) to be transferred to the test classification tasks, and the features used in the prediction need to be able to transfer knowledge across tasks (queries). This view is consistent with the *transfer learning* approach (Do and Ng 2005).

The traditional transfer learning approach aims to discover effective *parameter functions* that map training set statistics to model parameters, so that functions such as tf.idf that rely on heuristics and engineering may be discovered automatically. In the context of information retrieval, there is no existing parameter function for the prediction of effective term weights or $P(t|R)$, and such a function needs to be learned. Features that correlate well with effective term weights or with the term recall probabilities are used for predicting the target values. These features need to be general to adapt to different queries and different query terms in order to transfer knowledge about effective term weights or term recall probabilities from the training sets to the test tasks (queries). Some of these example features include the occurrence frequency of the query term in a query log (Bendersky and Croft 2008), the number of synonyms a query term has and the likelihood of a query term's synonyms appearing in place of the original query term in the collection (Zhao and Callan 2010).

## 2.6 Retrieval Techniques for Solving Term Mismatch

In the information retrieval literature, there are many different techniques that addresses term mismatch in one way or another. Since the dissertation research is about term mismatch, we summarize related prior techniques. We are mainly concerned with two points. The first is to distinguish the thesis research

from the prior research. The second point is to inform experimentation, which we take the following two paragraphs to explain in more detail.

Because these different techniques all address the same term mismatch problem, in a complex retrieval system, these techniques will interact with each other and cause a mixed effect that's hard to measure. Thus, it is important to point them out, so that in experimentation, we will understand their interactions and not be measuring the mixed effect of different techniques.

An example of interaction is, if no stemming is performed when evaluating the effectiveness of another technique that tries to solve term mismatch, e.g. query expansion, the effectiveness of the expansion method might be just because of its ability to discover the morphological variants of the original query terms, instead of other more interesting synonyms. Another example of interaction is that the effect of stemming measured on only retrieving the abstracts would most likely be more significant than its effect on retrieving the full text, because full texts usually already include multiple morphological variants of the query term, so stemming might be less in need.

In practice, when designing a retrieval solution or setting up a retrieval baseline, we should try to apply all of the following techniques to minimize the chance of mismatch.

### 2.6.1 Indexing and Document Representation

The representation of a document in the index can determine how likely query terms tend to match the document.

**From Keyword Indexing to Full Text Indexing**

During the early days of library science, books were indexed by a small set of keywords, and the search systems matched the query keywords only against the index keywords. The index terms form one representation of a document, which actually produces lots of mismatch for a naive user query. That's why librarians, experts that know the index system well, are introduced to help the naive user. Ever since the capability and speed of modern computers grew to be advanced enough to allow indexing of full text, full text indexing has been the norm, sometimes even replacing the manual labeling and indexing of documents. The switch into full text indexing significantly reduced the chances of query term mismatch, and the retrieval systems have been much more user friendly.

Similarly, *when documents are short, e.g. tweets, mismatch is much more likely to happen*. Thus, Twitter users insert or append hash codes to their tweets to facilitate search and reduce mismatch for commonly mentioned concepts. These hash codes behave just like the index terms in the library system but are created and used by the users instead of librarians.

**Inlink Anchor for Hypertext Documents**

Full text indexing is still not enough for solving mismatch. In the Web era, inlink anchor text, which is text from other hypertext documents used to describe the current document, has been used extensively to extend the original hypertext document, enriching it with keywords that did not appear in the document. See for example Chakrabarti et al. (1998) and Chapter 21 of the IR book by Manning et al. (2008). By extending the original document with more keywords, the chance of mismatch significantly reduces. User tags (e.g. those on Delicious.com) can similarly reduce term mismatch. The impact of using user tags to extend the text representation of the documents is more significant in the case of multimedia documents (pictures or videos) where very limited textual information is available in the original document to match the user queries.

**Query Log Based Document Content Enrichment**

User search logs, such as those collected by Web search engines or referrer URL information (Kemp and Ramamohanarao 2002; Scholer et al. 2004; Hillard and Leggetter 2010), can also provide the missing link

between a query term and a mismatched relevant document. The idea is that query terms used to search for a certain relevant document should be used to expand the representation of that relevant document.

In summary, all these different sources of information about the collection document should be considered a part of the document so as to avoid mismatch to the maximum extent.

### 2.6.2 Text Normalization

Another classical method that reduces term mismatch is stemming. Standard techniques include the Porter stemmer (Porter 1997) and the Krovetz stemmer (Krovetz 1993). By reducing morphological variants of a term to its basic form, the mismatch between query terms and document terms is easily reduced. In later developments, Tudhope (1996); Peng et al. (2007); Cao et al. (2008) and Dang and Croft (2010) all performed stemming in a query dependent way. By stemming according to the query context, stemming accuracy improves, and retrieval performance is also better than the query independent stemmers such as Porter and Krovetz.

In the special domains of bio medical or chemical search, researchers found that normalizing bio medical or chemical entity names using standard ontologies such as MeSH[5] or PubChem Substance[6] can be effective (Gurulingappa et al. 2010). For general domain full text retrieval, the use of ontologies does not consistently improve retrieval. Success is largely dependent on the careful handling of factors such as word sense disambiguation and query context, otherwise performance may be even worse than simple keyword retrieval. Automatic use of general ontologies such as WordNet is especially difficult, and most success cases come from using these ontologies to guide user interactions Bhogal et al. (2007).

### 2.6.3 Query Expansion

Compared to the above relatively simple and robust techniques, query expansion provides a more direct way to solve mismatch by including additional terms – expansion terms – into the query in order to match a larger number of the relevant documents. Query expansion itself is also quite attractive. As Rocchio (1971) put in one of the earliest work in this area, "search request formulation [...] is considered to be the variable with the most potential". This is because query formulation gives the system much larger control over the retrieval process and retrieval effectiveness, i.e. a more significant impact. Exactly in the same way is query expansion more challenging, as a bad formulation would be detrimental to performance, and stable expansion algorithms are hard to find.

Early on, researchers (Rocchio 1971; Robertson and Spärck Jones 1976) recognized that term distribution in the relevant set of documents for the query can be used to significantly improve retrieval. They used relevance feedback to improve query formulation. However sufficient relevance judgments are rarely available for a new test query. Over the years, considerable research has focused on pseudo relevance feedback (PRF), a method that assumes that the top ranked documents of an initial retrieval based on the original query are likely to be relevant (Croft and Harper 1979; Xu and Croft 1996; Mitra et al. 1998; Lavrenko and Croft 2001). These pseudo relevant documents are used to discover worthy expansion terms and proper expansion term weights, so as to better rank relevant documents. Prior pseudo relevance feedback methods did not distinguish the term weighting aspect of the method from the expansion aspect, but in theory the term weights estimated from pseudo relevant documents could be used to reweight the original query terms, as was done by Croft and Harper (1979) and this thesis research. This means pseudo relevance feedback methods are in theory more general than the term reweighting methods of Section 2.4.

[5] $http://www.ncbi.nlm.nih.gov/mesh/$
[6] $http://www.ncbi.nlm.nih.gov/pcsubstance$

Pseudo relevance feedback is on average very effective when evaluating over a set of queries, but performance on individual queries can vary considerably. It typically hurts performance when the top ranked documents from the initial rank list do not contain many relevant documents. Thus it is only occasionally used in operational environments. Collins-Thompson (2008) stabilized performance by performing sampling on query terms and top documents to give higher weights to stable feedback terms. The prevalence of a term in top returned documents is sort of a proxy to the prevalence in relevant documents, and stability resembles term recall. Computational costs are higher, but compared to previous query expansion methods, retrieval performance is more stable.

Below we review two different types of query expansion techniques. It is not meant to be complete, but more to show the different kinds of expansion query formulations out there, and to serve as the basis for the current thesis research.

#### 2.6.3.1 Bag of Word Expansion

Rocchio (1971) style expansion was one of the earliest types of expansion and is still fairly popular nowadays. Documents and queries are represented as bags of words with weights assigned to each word, conforming to the vector space model. The weights are usually the term frequencies in the corresponding query or document. The Rocchio algorithm was originally designed for the relevance feedback task, where given a query and its initial retrieval result set, the user would give a few judgments so that the system would know several example relevant and non-relevant documents. The algorithm simply does a weighted combination of the bag of words from the relevant examples, subtracting with the bag of words from the irrelevant class, and finally combining this set of feedback words with the original query. Since the feedback documents judged by the user will include terms that are not in the original query, the feedback query is often also called the expansion query. Again, as discussed in the section above, two factors in a bag of word expansion method contribute to the overall performance gain: 1) term weighting and 2) adding new terms (expansion terms) into the query.

When no relevance judgments are provided by the user, a pseudo relevant set, usually the top N documents from the initial retrieval, is used and assumed to be likely relevant. The same Rocchio style expansion can be used directly. This is called pseudo relevance feedback (PRF). Much prior research, and more recently TREC relevance feedback tracks (Lease 2010) have shown that simple PRF algorithms can bring a decent gain in overall retrieval performance. Since PRF algorithms do not require user feedback, i.e. they are automatic, PRF methods are applicable to more tasks than relevance feedback methods are. Experiments from the TREC 2009 relevance feedback track (Lease 2010) showed that when the number of relevant documents used in feedback is small, relevance feedback tends to be unstable, and PRF can improve the relevance feedback methods.

The dissertation research investigates two retrieval interventions, query term reweighting and query term expansion. For reweighting, we use pseudo relevance feedback information only to provide term weights for query terms; no expansion terms are included in the final query. For expansion, a structured form of expansion is used, which is discussed below.

#### 2.6.3.2 Structured/Per-Term Expansion

Although the bag of word expansion approach is simple and automatic, bringing nice accuracy gains on average, its behavior can be unpredictable for some queries. Expert searchers, on the other hand, prefer another form of expansion, the Conjunctive Normal Form (CNF) queries.

An example manual query from the TREC 2006 Legal track (Baron et al. 2007) created by lawyers for the e-Discovery task is as follows:

$$
\begin{array}{lll}
\#combine( & \#syn( & \text{guide* strateg* approval}) \\
 & \#syn( & \text{place* promot* logos sign* merchandise}) \\
 & \#syn( & \text{TV}\#1(\text{T V})\ \text{televis* cable network}) \\
 & \#uw5( & \#syn(\text{watch* view*}) \\
 & & \#syn(\text{child* teen* juvenile kid* adolescent*}))
\end{array}
\tag{2.12}
$$

This query is expressed in the Indri query language (Strohman et al. 2005), where query operator $\#combine$ means a probabilistic AND, $\#syn$ treats a set of terms as strict synonyms, $\#1$ is an ordered window of position different at most 1 (i.e. exact phrase), and $\#uw5$ is an unordered window of maximum size 5.

Overall, this expert created query expands the original query in roughly a Conjunctive Normal Form. This query looks for documents describing strategy or approval of cigarette signs or company logos on televisions that may be watched by children. Each concept or original query term in the query description is expanded to include additional terms that may be used in the document collection to refer to the same concept.

In two years of TREC Legal track experiments (Baron, Lewis, and Oard 2007; Tomlinson, Oard, Baron, and Thompson 2008; Zhu, Zhao, Callan, and Carbonell 2008), bag of word queries were never able to outperform the expert created Boolean queries. In fact, if we remove part of the structured query operators, so that structured queries deteriorate, the closer the structured query is to the expert created Boolean query, the better the performance is (Zhu et al. 2008).

Several earlier work also used manual Boolean queries in ad hoc retrieval and showed improvements (Hearst 1996; Mitra et al. 1998; Cormack et al. 1998). This CNF query formulation strategy has been called the *building block strategy* in library science early on (Lancaster 1968; Harter 1986).

Recent IR research is seeing a gradual increase in structured expansion or query reformulation, both in academia and in industry (Lamping and Baker 2005; Jones et al. 2006). These techniques appear partly because of the available query log data (query rewriting within a same session), and partly because structured querying gives finer control over which specific query terms to expand. Thus, structured query expansion or reformulation can be potentially more robust than the bag of word approach. Evidence from queries generated by expert searchers also tells us that structured queries provide more control over where to expand, and results in better retrieval performance.

More of the related manual and automatic structured expansion techniques are discussed in the semantic analysis section (Section 2.7) below.

### 2.6.4 Query Reformulation

Query expansion aims to solve the particular kind of term mismatch problems caused by synonyms of query terms. More generally, query reformulation techniques provide the framework to solve different types of mismatch problems. These reformulations include removing words from the query (Kumaran and Carvalho 2009; Dang and Croft 2010), adding new terms, replacing certain terms with others, and grouping terms that form a single concept together (Bendersky and Croft 2008). With the understanding of the term mismatch problem provided in this thesis research, the contributions of these different methods to solve mismatch can be made more explicit. For example, prior work included no analysis of whether a particular reformulation technique does in fact reduce mismatch, nor how much improvement in term recall is obtained.

**Query Term Reduction for Long Queries**
Query term reduction for long queries is a particular form of query reformulation that improves retrieval. The key observation is that for long queries, certain query terms are best removed to achieve better retrieval performance (Bendersky and Croft 2008; Kumaran and Carvalho 2009). These query terms tend to be unnecessary, as they represent unnecessary concepts, and should be removed from the query, so that the retrieval model will not prefer documents containing those terms.

**Query Term Substitution**
Query terms that represent necessary or high recall concepts, but are not effectively matching relevant documents can be replaced/expanded with better terms. Jones et al. (2006), Cao et al. (2008), Wang and Zhai (2008) and Dang and Croft (2010) performed automatic per term query expansion. In the case of Jones et al. (2006) and Wang and Zhai (2008) no end to end retrieval experiments were performed, thus, it is difficult to see whether the technique may improve retrieval. Cao et al. (2008) and Dang and Croft (2010) did not use stemming, and did not include inlink anchor texts for the baseline result, thus, it is difficult to judge how much of the retrieval performance improvement is due to expansion terms, and how much is due to pure stemming. Dang and Croft (2010)'s results also represent the oracle performance, varying the number of terms to expand per query, using the best performance for each query, and reporting the average across all queries.

### 2.6.5 Structured Retrieval Formalisms

As the section above shows, before the early 1990s, Boolean queries were widely used by search professionals, and unranked Boolean retrieval was popular. The research community switched to ranked retrieval much earlier from the 1970s and early 1980s (Salton et al. 1975; Robertson and Spärck Jones 1976). Around early 1990s, more and more research studies showed that unranked Boolean retrieval or strict interpretations of the Boolean operators is less effective than ranked retrieval (Lee and Fox 1988; Turtle 1994). The first commercial application of ranked retrieval was Westlaw's WIN system debuted in September 1992, which was based on Turtle and Croft's 1990 dissertation and the InQuery system. In 1994, web search engines started emerging, which were all ranked retrieval. Efforts were made to use soft or probabilistic interpretations of the Boolean operators to effectively use the Boolean queries for ranked retrieval. Earlier formalisms include the extended Boolean retrieval model (Salton et al. 1983). Later, the inference network is the first formalism to use structured queries in a probabilistic formalism, which was used first in the InQuery system and later in the Lemur/Indri search engine (Turtle and Croft 1990; Metzler and Croft 2004).

This dissertation research focuses on solving term mismatch, and one approach that this research takes to solve mismatch is structured queries in the form of Boolean Conjunctive Normal Form. Thus, this research relies on the structured retrieval formalisms to produce rank lists using Boolean queries. However, it is not the focus of this dissertation to compare structured retrieval formalisms, and we only adopt the inference network formalism in all of our experiments that involve Boolean queries. We believe other ranked Boolean retrieval formalisms that make reasonable assumptions about the retrieval process should behave similarly, as shown in earlier research (Lee and Fox 1988). More details of the ranked Boolean retrieval model is introduced in Chapter 8.

### 2.6.6 Retrieval Models

Several retrieval models directly aim to address term mismatch.

#### 2.6.6.1 Latent Semantic Indexing

Latent Semantic Indexing (LSI) (Deerwester et al. 1990) is a technique motivated by one of the earliest quantitative research on the vocabulary mismatch problem (Furnas et al. 1987). It aims to address term mismatch by representing the terms and the documents in a so called latent semantic space, and computes query-document similarity in that semantic space. Terms that are similar in the latent semantic space tend to be the terms that not only co-occur in collection documents, but also appear in similar contexts.

However, because of the computational complexity in doing Singular Value Decomposition over the term-document matrix of the whole collection, LSI is typically only applied to a sampled set of documents instead of the whole collection.

Schütze et al. (1995) extended the use of LSI by applying LSI on the top ranked documents from an initial retrieval. This localized the semantic space to the context of the current query, and improved accuracy. However, this local LSI method still does not show robust improvements over a keyword based method.

#### 2.6.6.2 Statistical Translation Based Models

Translation based retrieval models automatically model the vocabulary mismatch between queries and results documents through learning a model of word translations. The translation models approach vocabulary mismatch from the query's side and the documents' side. Significant improvements were observed over tf.idf baselines (Berger and Lafferty 1999). Training these translation models require a large enough query set to cover enough of the vocabulary, thus, it is key to gather enough accurate training data.

Gao et al. (2010) obtained large amounts of fairly accurate training data from user clicks in query logs, and used that data to train a translation based retrieval model to improve retrieval. However, there is a chicken-egg problem when using search engine results to predict mismatch or synonyms. If the search engine only return results that match all query terms, then it will be difficult to use user clicks to generate useful synonyms. This is because there are no mismatch cases between those query term and result document pairs used for training the translation model, so that the learnt model would not be able to extract useful synonyms for the original query terms.

#### 2.6.6.3 Least Square Learning of a Supervised Word-Concept Mapping

Yang and Chute (1993) introduced a technique that uses a set of training queries with relevance judgments to learn a word to concept (index term) mapping. The mapping is applied on test query terms to infer related concepts. The inferred query concepts are used to represent the query and to rank collection documents according to how close the documents are to the query concept vector in the concept space.

Better than the unsupervised latent semantic approaches, the least square method is supervised and can utilize relevance judgments in a training set to improve retrieval of a test query without relevance judgments. Different from the statistical translation models which learn a mapping between terms, the least square method relies on a set of canonical concepts (index terms) to learn a mapping from words to concepts.

This dissertation research also investigates a method trained with supervision. But different from the translation and the least square models that train a global translation model or a mapping between words which is applied uniformly on all test queries, this dissertation research shows that the best way to learn such a mapping between words is to do it in a query dependent way. We predict term weights and expand query terms in a query dependent way, so that the same term appearing in different queries may end up having different weights or different sets of expansion terms.

## 2.7 Prior Work on Semantic Analyses for Queries

Semantic representation of text provides a uniform and accurate representation of the meaning of the text, and ideally would give a perfect retrieval performance, solving the mismatch problem from the root. Many attempts have been made to index the meaning of the documents in the collection or analyze the meaning of the queries to improve matching.

At the same time, preprocessing the whole corpus is computationally and storage-wise expensive, especially when the optimal way of analyzing the texts is unknown. Because of that, and also because the query has a huge potential influence on retrieval performance, it is more practical to start such semantic attempts from the query's side.

Thesauri are frequently used to incorporate lexical semantic information in retrieval systems. An open domain thesaurus is hard to build, or if built, is hard to achieve a reasonable coverage across all domains (e.g. WordNet), and is hard to keep up to date. These thesauri are also static, trying to adapt them to individual queries and identify the right synonyms based on the query context is another important research question. However, in certain special domains where query terms have less ambiguity, e.g. bio-medical or chemistry, the use of pre-compiled thesauri or ontologies was proven very helpful, and is almost accepted as a norm (Hersh and Voorhees 2009).

We are more interested in open domain retrieval, and we review the following set of work that tries to use semantic information to formulate better queries for open domain full text retrieval.

### 2.7.1 Concept Grouping

The first step toward semantic understanding of the query or information need, is to identify the underlying set of necessary concepts. This could be done manually, or automatically, e.g. by grouping query terms into concepts.

#### 2.7.1.1 Manual

In library science, researchers and practitioners have long found the *building-block strategy* effective, where firstly, key (necessary) concepts are identified, secondly, each concept is represented as a disjunction (OR) of synonyms, and thirdly, the concepts are conjoined (AND) together to form the final query. The result is a conjunctive normal form (CNF) query, with each synonym group being one building block (Lancaster 1968; Harter 1986). The example query we borrowed from the Legal track is just one example of this form of Boolean queries. This query formulation strategy has also been widely used in text books to teach young professional searchers, with worksheets designed to specifically help searchers list relevant terms for each concept, so as to formulate high quality CNF style queries (Hensley and Hanson 1998). Compared to the original keyword queries, CNF-style (or per concept) expansion queries achieve higher recall because of the synonymous forms included for each concept. They can also maintain high precision because the result document must still contain all the queried concepts.

However, there are not many experiments behind these practices that compare whether these CNF queries do outperform bag of word queries. Researchers have compared Boolean search with keyword search, but most of those experiments assume that the Boolean system returns an unranked set of results, while in fact, the keyword system could provide the ranking needed by the Boolean system for a fair ranked Boolean retrieval vs. ranked keyword retrieval comparison. The fairer comparisons happen much later, by Hearst (1996); Mitra et al. (1998), and in TREC Legal tracks.

For ad hoc retrieval, Hearst (1996); Mitra et al. (1998) showed that *manually* grouping query terms from verbose (description) queries into concepts, and using the resulting CNF queries as Boolean filters

for keyword/ranked retrieval improves ranked retrieval significantly. Mitra et al. (1998) also extended the use of CNF Boolean filters to pseudo relevance feedback, showing that the Boolean filters stabilize pseudo relevance feedback (PRF) performance and substantially improves retrieval over both bag of word PRF and baseline original queries.

#### 2.7.1.2 Automatic

The success of the manual CNF query formulations suggests that automatic methods may do well. There is preliminary work showing some potential for this approach, but few concrete results exist. In terms of query concept identification and grouping, Zhao and Callan (2009) showed that automatically formulated Boolean (CNF) filters based on semantic role labeling of natural language questions significantly improves sentence retrieval for open domain question answering. Bendersky and Croft (2008) developed automatic key concept identification techniques showing a nice improvement on long queries, even though the final queries are not in CNF. Dang and Croft (2010) used per term expansion and showed that if the system knows how many expansion terms to include, a potential significant improvement over baseline retrieval is possible. Crabtree et al. (2007) tried to address under-represented query aspects in query expansion to avoid topic drift, but experiments used only 10 queries from the TREC 2005 HARD track queries, and the baseline is Google whose ranking algorithm is unknown and also changes frequently.

### 2.7.2 Searchonyms

After concept grouping, expansion terms still need to be identified for these concepts.

Searchonyms are terms such as synonyms, antonyms, hyponyms or even misspellings that "must be considered equivalent (to the query term) for search purposes", as coined by Richard P.C. Hayden (Lawlor 1962). Although legal and medical search professionals (Swanson 2003) have exercised the idea of expanding searchonyms into their search requests, not much research has been performed to understand how searchonyms could be identified either manually or automatically. Much research is needed to improve the understanding of searchonyms, what they are and how they can be identified to improve retrieval.

### 2.7.3 Other Semantic Analysis Work

Other forms of semantic analyses and query formulation have been tried by IR researchers over the years, although most of those happened in the early days, when people shared more hope to bring NLP techniques to IR. Most of these interventions turned out to be either inconclusive or unsuccessful for the retrieval task in general, except for some limited success in the question answering task. These attempts include Smeaton (1997) and Voorhees (1999), and are typically characterized by their mixed performance or ineffectiveness. Brants (2004) gives a survey for applying NLP tools in IR, reaching a similar conclusion.

A typical example of this kind of research tries to use NLP to generate additional information for the query and document keywords to further restrict the matching. For example, Brants (2004) reports the use of word sense disambiguation to increase the precision of the match between a query term and a document term by requiring the senses of the terms to be the same. Another example analyzes the syntactic structure of the query, and only matches documents that share the same structure. For example, Bilotti et al. (2007) matched the semantic role label structure of the query against those of the documents to restrict the match. These approaches improve the precision of the match, but only help a small number of the test queries. Success is difficult to observe. If not backing off the structure carefully, these approaches even decrease performance.

On the more successful side, Croft (1987) used "conceptual case frames" to formulate search queries, and showed initial success by manually creating the case frames for a small set of CACM queries. Zhu et al. (2006) used semantic relations between terms to resolve vague queries, however, no full scale experiments were conducted. Woods et al. (2000) aimed to solve mismatch by using a large manually created lexicon of lexical semantic relations among words, and actually showed that it helps improve retrieval. However, the experiments were not performed on any standard IR datasets. Though it included 90 queries, all of them are from only one user in a relatively narrow domain (UNIX). Thus, their approach does not seem scalable for more general applications. The interesting point, though, is that the use of thesauri was proved helpful outside just the bio-medical domain.

More recently, with long queries drawing more attention from the IR community, more NLP tools are being used and proved helpful. For example, noun-phrase chunking was used by Bendersky and Croft (2008), and Part Of Speech tagging was used to provide features (Lease et al. 2009). However, it is mostly not clear what benefit those NLP techniques have brought into the retrieval process. The dissertation research (Zhao and Callan 2010) shows that dependency parsing of queries can help predict term mismatch, and improve retrieval performance.

By studying the mismatch problem in contrast to improving the precision of the match between queries and documents, we are getting a clearer picture of where semantic analyses may benefit retrieval.

## 2.8 Query Difficulty and its Prediction

Query difficulty prediction (Cronen-Townsend et al. 2002; Carmel et al. 2006) is an active area of IR research that aims to automatically identify the lower performing queries that the retrieval model have problem dealing with. If successful, such predictions can guide interventions that try to address these problem queries, with term mismatch being one of the reasons for a query to have a poor performance.

Motivated by the failure analysis from the RIA workshop (Harman and Buckley 2009), Carmel et al. (2006) designed features to capture the agreement between query terms and the top returned results of the original query, and discovered that these features tend to correlate well with query difficulty. A small agreement between query terms and top results suggests a potential difficult query. This agreement between query terms and top returned results of the full query would be small if the top ranked results of the original query only emphasize a small number of the query terms, and ignore the rest of the query terms. As Carmel et al. (2006) pointed out, this is motivated by the emphasis problem (which is discussed in Section 3.6.1 in the dissertation). Carmel et al. (2006) used the features in a supervised setting to predict how likely a query is performing poorly, and showed that the predictions can benefit retrieval in several applications.

This dissertation shows that a low agreement between query terms and top ranked results of the full query is likely to occur when some query terms tend to mismatch relevant documents (Section 3.6.1). Thus, the features used to predict query difficulty could be predicting whether the query is suffering from the term mismatch problem. Indeed, two out of the three applications of the query difficulty predictions (Carmel et al. 2006) are about addressing the mismatch problem. One of the applications is to guide automatic query expansion, so that only difficult queries are treated with pseudo relevance feedback. Another application is to detect missing content where a query does not have any relevant documents in the collection. These observations suggest that perhaps the difficult queries identified there are those that tend to be suffering from the mismatch problem.

There is a large overlap between the query performance prediction methods and this dissertation research. Query difficulty prediction provides a way to identify queries that perform poorly, while we identify the mismatch problem which is a major problem in retrieval, especially for the kind of TREC

datasets used to evaluate query difficult prediction (Carmel et al. 2006). Thus, unsurprisingly, lots of hard queries where current retrieval models fail to return good results may also be the queries that suffer from the mismatch problem. The main difference between these two lines of research is that our term mismatch prediction provides more detailed diagnosis about each individual query term about whether the term suffers from the mismatch problem, while query difficulty prediction aims to decide for each query whether the query is difficult, regardless of what problems have made the query difficult. The more detailed diagnosis provided by term mismatch prediction makes it easier to apply the diagnosis to guide automatic or interactive retrieval interventions to improve retrieval in more principled ways.

## 2.9 Summary

Term mismatch is a long standing problem in information retrieval, drawing attention early on. Though without formalizing and quantitatively studying the mismatch phenomenon, the research community has developed various techniques trying to address the mismatch between query terms and relevant documents, all of which informs the dissertation research and future research on modeling and solving mismatch in retrieval. One observation from these prior retrieval techniques is that techniques that can address term mismatch tend to be the techniques that result in the most reliable retrieval accuracy gains. Techniques that are precision based, e.g. dependency models or using syntactic structures to restrict matching, are not as reliable, e.g. they tend to improve top precision at the cost of decreasing precision at lower ranks, or show less reliable improvement over the baseline keyword retrieval. An explanation of this observation comes in the chapter below.

The dissertation research distinguishes itself from the rest by adopting an empirical and quantitative approach to the mismatch problem. We define the mismatch probability and perform exploratory data analyses to investigate the causes of mismatch, and to further inform experimentation. We point out that term mismatch probability is an important part of optimal term weighting, and that it can guide the formulation of effective queries. As probably the first attempt to investigate the term mismatch phenomenon in retrieval, we expect our findings to contribute a significant amount of knowledge to the information retrieval field, in analyzing, predicting and solving term mismatch in retrieval.

The chapter below explains in more detail how term mismatch relates to ad hoc retrieval models and retrieval performance.

# Chapter 3

# The Term Mismatch Problem & its Causes

## 3.1 Introduction

Researchers have long recognized that the vocabulary mismatch between user query terms and collection documents is an important problem for information retrieval or human computer interaction (Furnas et al. 1987). These mismatched relevant documents, potentially a very large portion of all the relevant documents for the query, are usually poorly ranked by common retrieval models and techniques. However, prior approaches focused more on solutions, for example query expansion techniques, without first carefully studying how and why query terms mismatch relevant documents for the query.

One contribution of this research is the recognition that the probability $P(\bar{t}|R)$ measures term mismatch. Thus, a standard retrieval dataset with queries and relevance judgments can be used, without any additional manual labeling, to study the term mismatch phenomenon objectively and quantitatively. In particular, it is possible to measure for different query terms how much mismatch probability varies, to find out what kind of terms are likely to mismatch, what causes mismatch, whether there is variation for the mismatch probability of the same term occurring in different queries, and why.

Overall, this new perspective provides new insights into the age old mismatch problem, enabling principled interventions to be designed to predict term mismatch, solve mismatch and improve retrieval.

In this chapter, a set of exploratory data analyses (EDAs) are designed to answer the following questions regarding the term mismatch probability.

1. How much does the term mismatch probability vary across different query terms? Or, what are the characteristics of the distribution of the term mismatch probability across query terms? This analysis intends to show whether modeling and predicting of the term mismatch probability is needed in the first place.
2. Are there significant differences between the distributions of the term mismatch probability for different types of queries (long vs. short) or different kinds of document collections (small vs. large, newswire vs. Web)? This tries to investigate the prevalence of the mismatch problem, and applicability of the techniques that try to address term mismatch.
3. How much does the mismatch probability of the same term vary across different queries? This question aims to find out whether the use of query dependent features is needed for mismatch prediction.
4. How does the term mismatch problem affect retrieval effectiveness? The goal is to discover the underlying mechanism of how ignoring term mismatch in retrieval affects retrieval performance. The identified pathology of the term mismatch problem will suggest effective interventions to solve

Table 3.1: TREC datasets used for testing.

| TREC track | Ad hoc track | | | Web track | | |
|---|---|---|---|---|---|---|
| TREC dataset number | 4 | 6 | 8 | 10 | 12 | 14 |
| TREC year | 1995 | 1997 | 1999 | 2001 | 2003 | 2005 |
| Total #documents | 0.568M | 0.556M | 0.528M | 1.692M | 1.248M | 25.205M |
| Avg doc_length (#terms) | 533 | 556 | 510 | 636 | 1124 | 717 |
| Total #queries | 50 | 50 | 50 | 50 | 50 | 50 |
| #judgments per query | 1741 | 1445 | 1737 | 1408 | 1021 | 906 |
| #relevant_docs per query | 130 | 92 | 95 | 67 | 10 | 208 |

the problem in retrieval.

5. What causes query terms to mismatch relevant documents? This analysis is necessary to discover patterns in all the different cases of term mismatch, and to guide the design of effective features to predict the term mismatch probability.

## 3.2 Datasets

Before describing any empirical results, we describe the datasets we use to analyze and evaluate term mismatch prediction. 12 standard TREC ad-hoc retrieval datasets are used (TREC 3-14), 6 of which (TREC 3, 5, 7, 9, 11 and 13) are used as training sets, and the rest are used as test sets. We looked at only training set queries and relevant documents for data analyses.

Smaller datasets with more complete judgments include the ad-hoc retrieval tracks of TREC 3 to 8. Larger datasets are Web tracks of TREC 9 and 10, Topic Distillation tasks of TREC 11/2002, 12/2003 and Terabyte tracks of TREC 13/2004 and 14/2005. The larger datasets also have sparser judgments. Table 3.1 lists the statistics of the test sets used in the experiments.

### 3.2.1 Query Characteristics Across Query Sets

The $P(t|R)$ probability of a term in a short query tends to be higher than the $P(t|R)$ of a term in a long query. For example, TREC 3 titles have a similar verbosity level as TREC 9 descriptions, with an average of 5-6 terms per query. TREC 4-8 descriptions are more verbose, averaging 8 to 9 terms per query. TREC 4-8 descriptions have an average term recall of 0.38 to 0.43, while TREC 3 titles and TREC 9 descriptions have a higher average recall of 0.54 to 0.59.

Despite the change of characteristics of TREC queries from year to year, we show in the chapters below that training data from one TREC dataset can be used to predict $P(t|R)$ for other datasets.

### 3.2.2 Text Representation

The Krovetz stemmer was used for parsing both queries and documents. Without stemming, the unstemmed query term matches fewer relevant documents, and leads to lower recall.

#### 3.2.2.1 Document Processing

For document processing and indexing, stopwords are left in.

For Web documents, anchor texts from in-links are included as part of the content, which improves term matching and increases $P(t|R)$.

#### 3.2.2.2 Query Processing

Meta-language terms in the queries, e.g. instances, discuss, relevant etc., are generally not useful. We used simple rules to remove those phrases that the queries begin with, e.g. "(find | identify | provide) (documents | reports | information) (on | about)". Removing these improves baseline ad-hoc retrieval performance, but has little effect on runs that use term recall based query term reweighting.

As for stopwords, only prepositions (e.g. in, to, on, beneath, or "instead of" etc.) and conjuncts (and, or, not etc.) are removed, which is caused by the behavior of the Stanford dependency parser which collapses these dependencies. Other stopwords such as determiners, are left in.

## 3.3 Estimation Using Relevance judgments: Training Data Generation

For both exploratory data analyses and supervised prediction of term mismatch/recall[1], ground truth probabilities, or estimates of $P(t|R)$, are needed.

For a given query $q$, $R =$ the set of relevant document for $q$, $r =$ the set of relevant documents for $q$ that contain term $t$.

Then, *true term recall* can be computed from the set of relevant documents directly. By definition,

$$P(t|R_{full}) = \frac{|r_{full}|}{|R_{full}|} \tag{3.1}$$

where —$R_{full}$— is the cardinality of the set $R_{full}$, i.e. the total number of relevant documents for $q$.

In the above calculation, the assumption is that the full relevant document set is known. In reality, complete relevance judgments are seldom available for any reasonable sized document collections. Thus, true term recall is seldom known, and we can only rely on the set of known relevant documents as a sample from the full set to *estimate* recall. An unbiased estimator of term recall is just the probability calculated on the sampled relevant documents,

$$P(t|R_{sample}) = \frac{|r_{sample}|}{|R_{sample}|} \tag{3.2}$$

where $r_{sample}$ is the set of known relevant documents that contain $t$, and $R_{sample}$ the set of documents known to be relevant to the query.

For some queries, the set of known relevant documents is small, resulting in a small $|R_{sample}|$ in the denominator, and a large variance in the final estimated recall probabilities. A standard solution is to use smoothing to reduce the variance. We use *Laplacian smoothing* in all our experiments. The smoothed estimator is,

$$P_s(t|R) = \frac{|r_{sample}| + 1}{|R_{sample}| + 2} \tag{3.3}$$

Laplacian smoothing is also called "add one" smoothing. It assumes that before looking at the relevant documents from the empirical data set, *a priori*, we have already seen two relevant documents, one containing the term $t$ and one that does not.

[1]Remember that by definition, the term recall probability is the complement of the term mismatch probability; they sum up to 1.

Table 3.2: Different types of corpora and queries used in the exploratory data analyses.

| Corpus | Description | Total #docs | Avg doc length (words) |
|---|---|---|---|
| TREC 3 ad-hoc | Newswire | 0.568M | 533 |
| wt10g (TREC 9 Web) | Web | 1.692M | 636 |
| GOV2 (TREC 13 Terabyte) | Web | 25.205M | 717 |
| **Queries** | **Query field** | **Total #queries** | **Avg query length (words)** |
| TREC 3 ad-hoc | Title | 50 | 4.9 |
| TREC 9 Web | Title | 50 | 2.5 |
| TREC 9 Web | Desc | 50 | 6.3 |
| TREC 13 Terabyte | Title | 50 | 3.1 |
| TREC 13 Terabyte | Desc | 50 | 6.9 |

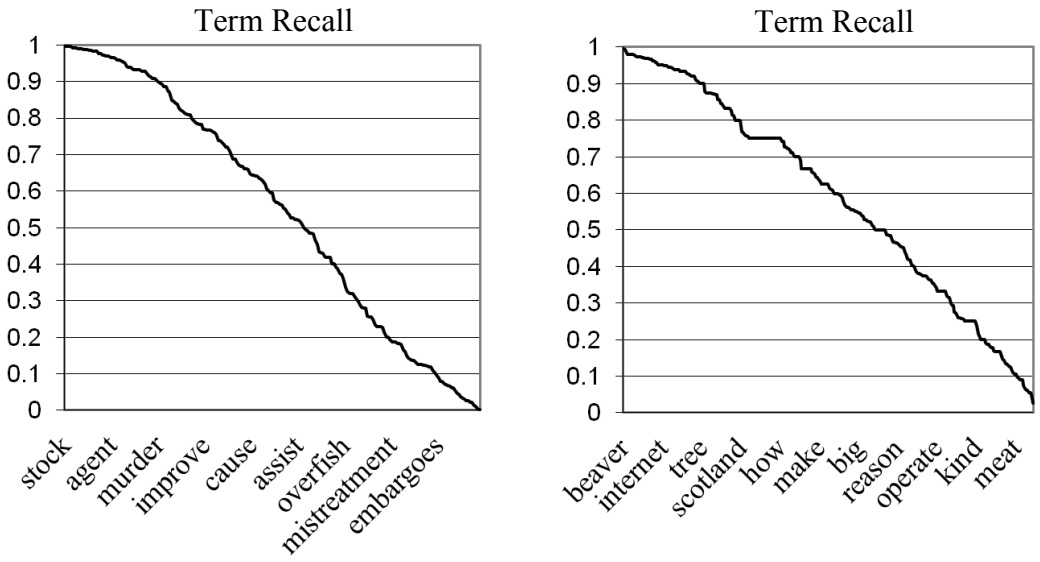
Pooling is a technique widely used in IR evaluation to save assessor effort while still generating a reusable test collection. During pooling, only top ranked documents from a set of retrieval runs are manually assessed for relevance. Using pooled judgments might bias the estimation of term recall, because top ranked relevant documents are typically the easier ones to retrieve, containing most of the query terms, while lower ranked relevant documents that have never entered the pool would be more likely to mismatch lots of the query terms. This dissertation's work on multiple TREC datasets shows that sparse judgments (the larger Web track collections in Table 3.1) give a similar amount of improvement in retrieval performance (Table 6.1), which suggests that $P(t|R)$ can still be reasonably effectively estimated on pooled judgments.

## 3.4 Large Variation of Mismatch Probability for Query Terms

Basic retrieval models such as Okapi BM25 or the query likelihood version of the Statistical Language Models assume that the term mismatch probability across all query terms stays the same. The exploratory data analysis (EDA) by Greiff (1998) showed otherwise, that the term mismatch probability can range widely from 0 to 1. However, nobody has really examined the variation or characterized the distribution of the term mismatch probability across different query terms. The following EDA does that by showing the term recall probability distribution. This will help us understand the probability we are trying to model, as well as its importance. This kind of distribution analysis is also practically important, as a sanity check, to decide whether to apply mismatch prediction techniques to a new domain, a new retrieval task, or even just a new set of queries.

As shown in Table 3.2, different types of collections and different types of queries are investigated to show that term mismatch is a general problem. True term recall probabilities are estimated from relevance judgments as the proportion of relevant documents that contain the term. Laplacian smoothing (a.k.a. add-one smoothing) is used. The query terms and document text are stemmed with the Krovetz stemmer that comes with the Lemur toolkit. Some minimal stopword removal is performed. More details about text representation/preprocessing can be found in Section 3.2.2. Details about term recall estimation can be found in Section 3.3.

Figures 3.1a and 3.1b show the recall probabilities of different query terms from multiple TREC datasets. TREC queries typically consist of a title (several keywords), a description (a one sentence summary) and a narrative field (several sentences narration of the information need). The earlier TREC tracks had much longer titles than the later Web tracks. In Figure 3.1a, although TREC 3 title fields are




(a) TREC 3 query term recall in descending order, for title queries (averaging 4.9 terms per query over 50 queries). Representative terms are shown on the x-axis.

(b) TREC 9 query term recall in descending order, for description queries (averaging 6.3 terms per query over 50 queries). Representative terms are shown on x-axis.

Figure 3.1: Term recall ($P(t|R)$) for long queries from two TREC datasets.

used, they are on average 4.9 terms long, almost as verbose as the TREC 9 description fields, which has an average 6.3 terms.

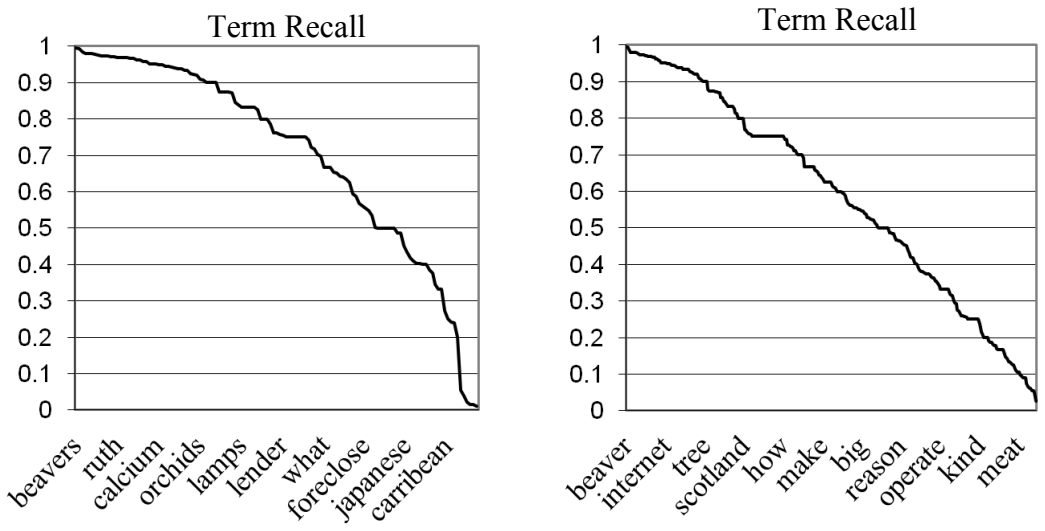
The Figures 3.1a and 3.1b show a similar trend in recall variation. The distribution of term recall probabilities is almost uniform from 0 to 1, definitely not constant. There is a similar number of high recall terms as there are low recall terms. Despite being from the title field of TREC 3 queries, the term recall distribution is very similar to that of the TREC 9 description queries.

Bendersky and Croft (2009) defined long queries as queries 5 to 12 words long in a study of a large Web search engine log. In other domains, such as legal or patent search, queries are typically much longer; lawyers would not think of a 5 word query as a long query. In this thesis, when we talk about long queries, we follow the Web search practice of 5 or more words. The reason mainly is that we are using ad hoc and Web track queries and datasets from TREC which is closest to the Web search domain.[2]

### 3.4.1 Short/Keyword Queries vs. Long/Verbose Queries

In Figure 3.2a, we present the recall probabilities for TREC 9 title queries, to examine whether term mismatch is a severe problem for shorter queries. The figure shows that the term recall probabilities for short title queries still vary from 0 to 1, occupying the full range. This shows that term mismatch is still quite prevalent for shorter queries. When compared to the description queries (Figure 3.2b), the title queries have a clear skew toward higher recall, i.e. there are more high recall terms than there are low recall ones. Results on short queries of TREC 13 (Figure 3.3a below) show similar trends. This suggests that even for short queries, retrieval models that are aware of term mismatch could still improve over

[2]Another reason behind this definition of long queries is related to the emphasis problem (Section 3.6.1). With 5 or more words in a query, it is very likely there will be 1 or 2 low recall terms that will cause the retrieval models to incorrectly bias toward them, causing the emphasis problem. Thus, techniques that try to solve mismatch would likely see easier improvements over queries that are 5 or more words long. For fewer words, the mismatch problem would likely be less severe.




(a) TREC 9 query term recall in descending order, for title queries (averaging 2.5 terms per query).

(b) TREC 9 query term recall in descending order, for description queries (averaging 6.3 terms per query).

Figure 3.2: Term recall ($P(t|R)$) for short queries, compared with the long query version.

baseline models (such as Okapi BM25) that only use idf based term weighting, although the improvement may not be as significant as on description queries. This is also confirmed by term weighting experiments in Chapter 6.

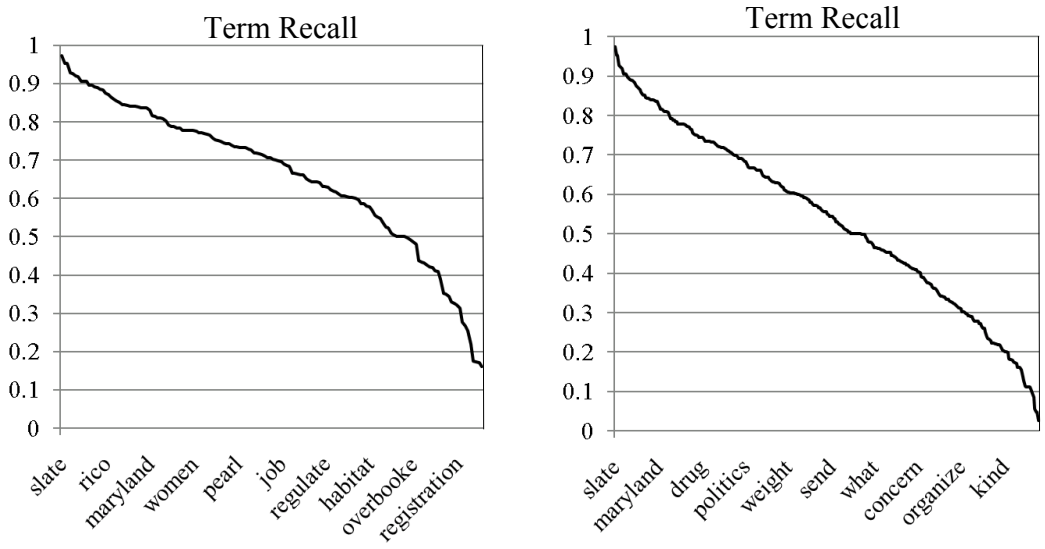
Because of the larger variation in mismatch probabilities for long queries, we can expect mismatch prediction based retrieval models to improve more over the baseline retrieval models on long queries than on short queries. This is confirmed by term weighting experiments in Chapter 6.

### 3.4.2 Small/Newswire vs. Large/Web Collections

Figures 3.3a and 3.3b show the distribution of the term recall probability for short vs. long queries on the GOV2 collection of about 25 million Web pages. Firstly, even for short queries on a large document collection, there is still a fairly large variation of the term mismatch probability, ranging from 0.16 to 1. 20% of the query terms still mismatch more than half of the relevant documents for the query. Description queries have a similar term recall distribution as those from much smaller collections (TREC 3 or 9). Secondly, shorter queries do generally increase term recall values. The bias to higher recall is even more evident than that on the TREC 9 title queries.

Overall, query characteristics (the lengths of the queries) seem to affect term recall distribution the most. The average document length and the size of the collection do not seem to affect the term recall distribution in any significant way.

Because of the large size of the Web collections, when using pooling to do relevance assessments, pooling on such a large corpus is even more likely to miss relevant documents than pooling on a relatively small collection. Because undiscovered relevant documents would typically mismatch lots of the query terms, these plotted term recall values are likely to be over-estimates.




(a) TREC 13 query term recall in descending order, for title queries (averaging 3.1 terms per query).

(b) TREC 13 query term recall in descending order, for description queries (averaging 6.9 terms per query).

Figure 3.3: Term recall ($P(t|R)$) for short and long queries on GOV2 (a large Web document collection).

Table 3.3: Term recall ($P(t|R)$) of 5 example terms (stemmed) on 5 sample queries from TREC 3 title queries.

| **Query (TREC 3 Ad Hoc Retrieval Track)** | **Term** | $P(t\|R)$ | **idf** |
|---|---|---|---|
| Oil *Spills* | spill | 0.9914 | 5.201 |
| *Term* limitations for US Congress members | term | 0.9831 | 2.010 |
| Insurance Coverage which pays for Long *Term* Care | term | 0.6885 | 2.010 |
| School Choice Voucher System and its *effects* on the US educational program | effect | 0.2821 | 1.647 |
| Vitamin the cure or cause of human *ailments* | ail | 0.1071 | 6.405 |

### 3.4.3 Examples of Mismatched Query Terms

True term recall probabilities for sample terms from TREC 3 title queries are shown in Table 3.3. Because prior research used idf to predict term mismatch, idf values ($log(N/df)$) are also listed for reference.

Table 3.3 shows that $P(t|R)$ is not constant across different query terms, nor is it constant for a specific query term. For example, the word 'term' has high recall for one query (0.9831), but lower for another (0.6885).

Table 3.3 also shows that there is no simple correlation between $P(t|R)$ and idf. On a larger scale, Figure 3.4 and Greiff's (1998) data analyses further confirms the point.

Overall, these results suggest that the term mismatch problem is prevalent in ad-hoc retrieval, even for short queries and large collections. This is the first evidence that directly motivates the prediction of the term mismatch probability. Results also show that idf alone is not enough for modeling and predicting term mismatch.

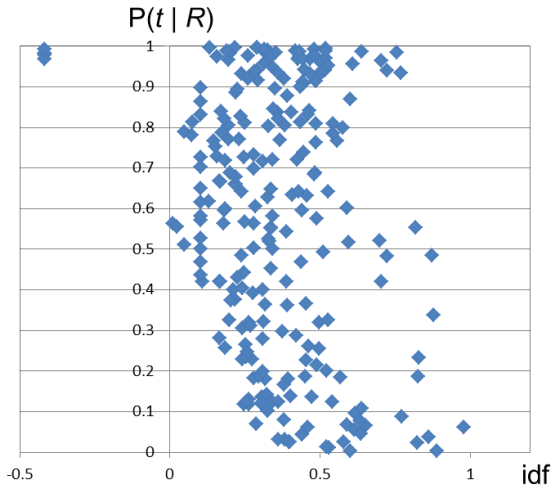



Figure 3.4: Scatter plot of TREC 3 description query terms, with $x$-axis being idf, and $y$-axis term recall.

## 3.5 What Causes Mismatch

One main contribution of this work is to point out some of the factors that might cause term mismatch or affect term recall, to motivate later design of features to recognize or model it.

We start our analysis by looking at some of the highly mismatched terms. We use TREC 3 queries for data analysis, which we only use as training data in our experiments. We have discovered a set of factors that cause high term mismatch or affect term recall, which are listed below.

1. If a term is not **central** to a query, it is unlikely to have high recall. Here we assume that terms occurring consistently in top documents are central. For the query "Vitamin - the cure of or cause for human ailments", "ailments" appears less consistently in top documents than vitamin, and in fact has a much lower recall of 10.71% than "vitamin" of 96.43%.
2. If a term does not represent a **central concept** for the query, then the term in turn would likely have a low recall. Examples of concepts that are not central to the query include "potentially related" in "Laser research related or potentially related to US defense", or "propounded" in "Welfare laws propounded as reforms". Some queries may contain meta-language concepts that are not helpful for retrieval, e.g. "Find me documents about".
3. Synonyms or other related terms such as antonyms or hyponyms may **replace** the original query term in relevant documents, lowering the recall of the original query term. In "US educational system", if relevant documents say "America" or "USA" instead of "US", it lowers the recall of "US". For terms with multiple senses, only the synonyms of the queried sense should matter. If a term does not often co-occur with its synonyms, then it is often replaced by those synonyms, and the term would typically have a low recall. Lu et al. (2009) used a similar measure, synonym novelty, to assess how many new documents a synonym can match that the original query term cannot.
4. Many **abstract** but **rare** terms typically have low recall, because they are at a different level of abstraction than the relevant documents, even though the concept that the term represents could be important. Examples are "maulings" in "dog maulings", "fundamentalism" in "Christian fundamentalism" and "ailments" in the vitamin query. Often, hyponyms of abstract terms appear in relevant documents.

Table 3.4: Idf and recall for query terms from the query "prognosis/viability of a political third party in the US" (TREC ad hoc track query 206).

| **term** | **party** | **political** | **third** | **viability** | **prognosis** |
|---|---|---|---|---|---|
| **idf** | 2.402 | 2.513 | 2.187 | 5.017 | 7.471 |
| **True** $P(t\|R)$ | 0.9796 | 0.7143 | 0.5918 | 0.0408 | 0.0204 |

It is important to note that term recall is different from concept recall. "maulings" is a high recall (important) concept for the "dog maulings" query, but the term "maulings" does not have a high recall, because of searchonyms such as "attack" or "bite".

## 3.6 Term Mismatch and Retrieval Performance

Theory predicts that term mismatch/recall should have a strong influence on retrieval performance. It has been known since the 1970s that $P(t|R)$ is the only variable unaccounted for for optimal term weighting (Robertson and Spärck Jones 1976).[3] $P(t|R)$ is also effectively the only variable about relevance, because $P(t|\bar{R})$ is accurately approximated by idf, thus is not dependent on relevance (Greiff 1998). This is true for any retrieval model that only uses idf based term weighting, which includes BM25 and query likelihood language models that also use term frequency (tf) and document length information.

Empirically, Robertson and Spärck Jones (1976) demonstrated that simply weighting query terms according to their true recall values produced 80% to 100% improvement in precision at all recall levels over the predictive Binary Independence Model. Greiff (1998) confirmed this result using predicted term weighting against the same simple BIM baseline on early TREC datasets. Our recent experiments (Section 6.2.3) confirm that 30% to 80% improvement is possible with more modern retrieval methods, higher baselines, and much larger test collections.

In this section, we explain how term mismatch impacts retrieval performance, especially how term mismatch can cause the emphasis and the mismatch problems, potentially hurting retrieval accuracy at all rank levels. In particular, a method that addresses term mismatch can improve both retrieval precision and recall.

### 3.6.1 The Emphasis Problem and its Pathology

Retrieval models with idf-based term weighting will prefer matches of high idf terms over the matches of the other equally necessary but low idf query terms. This causes retrieval models to bias their top ranked results to only emphasize some aspects of the query but not all of the necessary aspects. We call this the *emphasis* problem.

For example, for the query "prognosis/viability of a political third party in the US", *prognosis* and *viability* have the highest idf (as listed in Table 3.4), but if considering both idf and recall, *political third party* should instead be emphasized.

Table 3.5 shows the snippets of the top ranked results for the query "prognosis/viability of a political third party in the US", using language modeling retrieval model with Dirichlet smoothing.

The wrong emphasis causes a serious problem: false positives (irrelevant documents that happen to match part of the query) appearing throughout the rank list.

[3]Note that even though BIM is an old model, it is not obsolete. It appears as the term weighting component of the more advanced Okapi BM25 model, and term weighting based on tf.idf is prevalent in almost every retrieval model.

Table 3.5: The snippets for the top 10 ranked documents for the query "prognosis/viability of a political third party in the US", on the TREC 5 dataset, using language modeling retrieval model with Dirichlet smoothing.

| | |
|---|---|
| 1. | ... discouraging prognosis for 1991 ... |
| 2. | ... Politics ... party ... Robertson's viability as a candidate ... |
| 3. | ... political parties ... |
| 4. | ... there is no viable opposition ... |
| 5. | ... A third of the votes ... |
| 6. | ... politics ... party ... two thirds ... |
| 7. | ... third ranking political movement... |
| 8. | ... political parties ... |
| 9. | ... prognosis for the Sunday school ... |
| 10. | ... third party provider ... |

For example, some false positives are about "prognosis of [the next year]", or "viability of [a person]'s candidacy", or "prognosis" in one place of the document but "third party [supplier]" in another part of the document.

These false positives contain some but not other necessary aspects of the query, and are especially detrimental when they appear at top ranks. All of the top 10 results from a search by the Indri retrieval system on the TREC collection are false positives, symptom of the emphasis problem. Several of the top ranked documents are about prognosis or viability of something else, and political third parties do not appear in them.

Even on Google or Bing, the emphasis problem is still observable. Within the top 10 results, there are two false positives (Figures 3.5 and 3.6).

The retrieval models assigning the wrong emphasis to certain query terms is an effect that is most observable when the low recall query terms also have high idf values. But in general the emphasis problem could hit equally badly those queries with low idf and low recall terms.

One thing quite contrary to common intuition is that the problem suffered by this query is not a term precision problem. If just looking at the top ranked false positives, it is very natural for someone trying to improve the query to eliminate the top false positives by simply requiring "political third party" to appear as a phrase or "prognosis" to appear close to "political third parties". However, for this query, requiring more precise matching would only make the resultant query even more likely to mismatch relevant documents (lower recall), and further decrease performance, or for some queries it may increase top precision but decrease precision at lower ranks. For this query, term mismatch and emphasis is more of a problem.

The wrong emphasis causes false positives to appear throughout the ranked list, decreasing both precision and recall. Proper term weighting, e.g. based on term recall, will not only improve retrieval recall, but also precision.

### 3.6.2 The Mismatch Problem

The mismatch problem is a closely related problem caused by relevant documents not containing a certain query term, resulting in a poor ranking for such relevant documents.

The mismatch problem is a more fundamental problem than the emphasis problem. It is a root cause of the emphasis problem. The emphasis problem can occur only when at least one of the query terms has


Figure 3.5: Google's top ranked results for the query "prognosis/viability of a political third party in the US". The last two results are false positives. Result page accessed in October, 2010.

a high probability to mismatch relevant documents (i.e. has a low recall).

The mismatch problem also has a more significant impact on retrieval performance. The performance gain achievable by correcting the emphasis problem using term weighting is just a lower bound of the possible gain achieved from solving term mismatch. Specifically, term weighting can only deemphasize the highly mismatched terms, but will not be able to rank highly those relevant documents that only contain a synonym of the original query term. However, if a complete and accurate set of expansion terms are included in a disjunction with the original query term, the final disjunction would match most of the relevant documents, bringing those previously mismatched relevant documents to the top, increasing recall and precision at every rank, and solving both the emphasis and the mismatch problem.

Even though an accurate and complete set of expansion terms/searchonyms for each query term is hard to get, this way of solving the mismatch problem by expanding each term individually suggests the

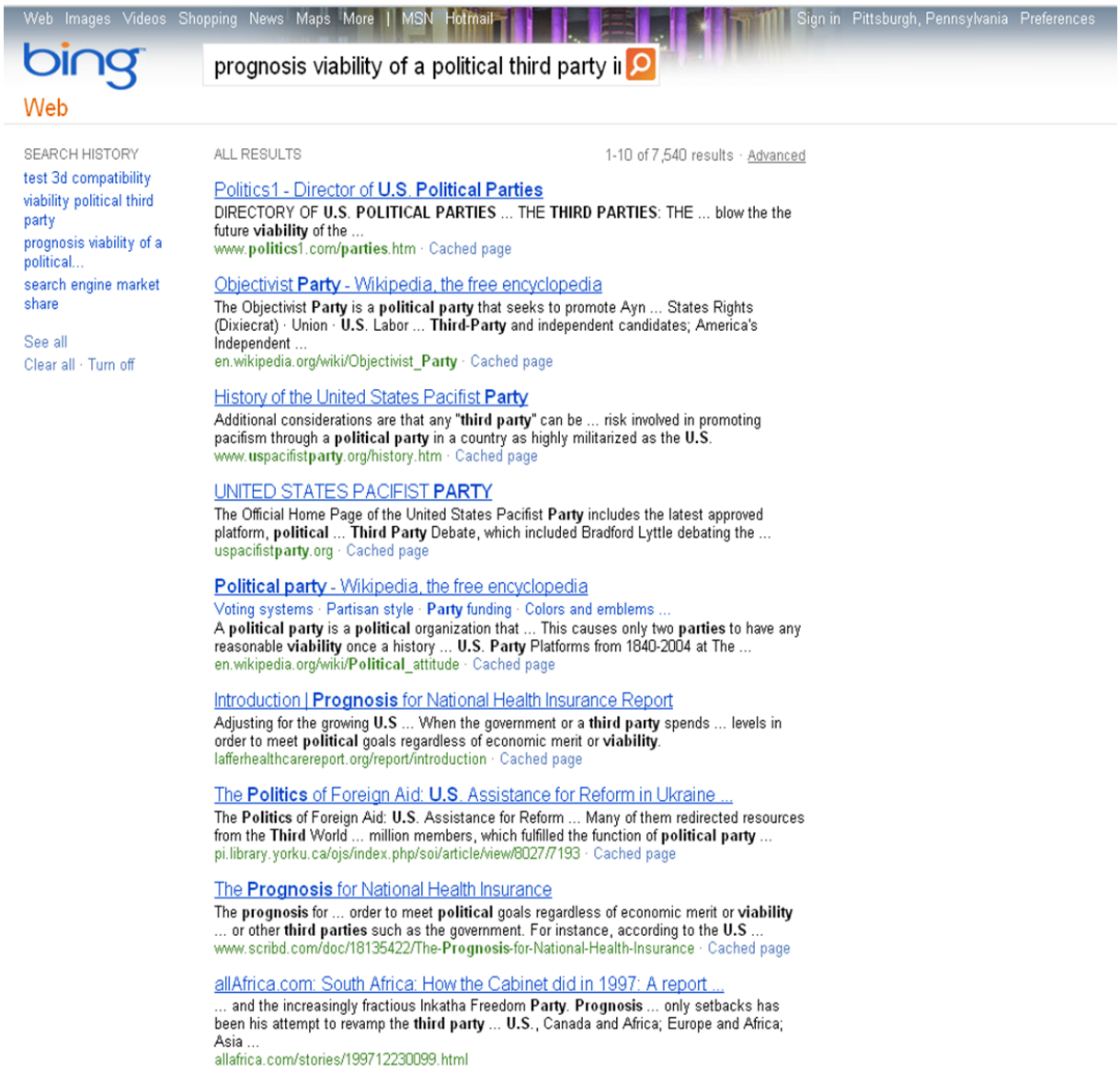



Figure 3.6: Bing's top ranked results for the query "prognosis/viability of a political third party in the US". Result number 6 and 8 are false positives. Result page accessed in October, 2010.

use of Boolean Conjunctive Normal Form (CNF) queries.

When the emphasis and mismatch problems occur, both retrieval precision and recall suffer. But how often do they happen? In prior research, the large potential performance gains due to using true recall term weights suggested indirectly that the emphasis problem is prevalent. We show more direct evidence below.

### 3.6.3 Failure Analysis of the State of the Art

Failure analyses of state of the art systems allow researchers to identify the causes of the failures, and also the significance of each cause. For this reason, the 2003 Reliable Information Access (RIA) workshop

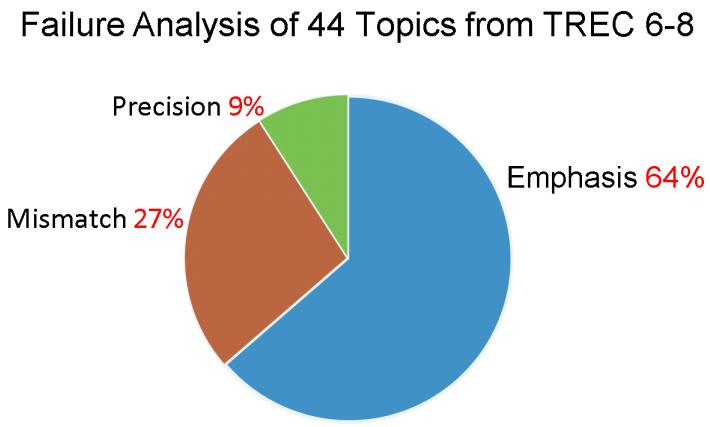



Figure 3.7: A breakdown of the causes of failures for common IR techniques in ad-hoc retrieval – a summary of the analyses by the 2003 RIA workshop (Harman and Buckley 2009).

(Harman and Buckley 2009) served the IR community well by spending weeks of time analyzing the failures and causes of failures of 7 state-of-the-art research search engines and common IR techniques. However, the results of the RIA workshop have been difficult to interpret, and the community has been unable to apply the results to guide retrieval research. We provide a summary of their analyses and an interpretation which argues for term mismatch as the area of information retrieval that needs most attention.

The RIA workshop provided a detailed failure analysis for 44 queries where most of the 7 top research retrieval systems performed poorly. Their causes of failure are described in natural language. We categorize all the different causes into 3 categories. This categorization is directly carried out based on the natural language descriptions of the causes of failures.

1. The *emphasis* problem (64% of the failures): The retrieval systems all emphasized only some aspects of the query (the aspects containing mostly high idf query terms), while missing other "required" aspects/terms. An example is "the prognosis/viability of a political third party", where retrieval systems would prefer a match of the high idf terms "prognosis" or "viability" but not "third" or "political" or "party". For a majority of these failed queries, all the retrieval systems tested in the workshop failed in the same way, emphasizing one specific part of the query. This means the failures are systematic, not coincidences. This emphasis problem can be alleviated by proper recall based term weighting.
2. The *mismatch* problem (27% of the failures): Queries needed expansion of a general term, e.g. "Europe", or a difficult aspect e.g. "new methods ..." and "mercy killings" (which should also match "right-to-die"). Accurate term mismatch predictions would guide the expansion algorithm to the terms for which mismatch most likely happens and expansion is most in need.
3. The *precision* problem (9% of the failures): Queries needed either syntactic relations between query terms to disambiguate them or needed lexical disambiguation e.g. "euro" as a currency instead of "euro-something".

Emphasis and mismatch problems account for 91% of the failures in RIA queries. Furthermore, these failures are systematic, not specific to a particular system or technique; all the tested systems fail for the same reasons, except for only 5 queries where different systems emphasize and miss different parts of the query (Harman and Buckley 2009, Section 3.5). This is direct evidence that term mismatch is an urgent

and impactful problem to solve in retrieval, and that a deeper and systematic reason may be the underlying cause.

### 3.6.4 Why Much Attention is Given to Term Precision, instead of Term Recall or Mismatch?

Given that the term mismatch problem is so important, why is the term precision problem receiving so much attention, instead of term recall and mismatch? Why did so many prior attempts aim to improve retrieval through increasing term precision, using e.g. natural language processing and various other techniques? Prior research has also found that people often underestimate recall problems (Blair and Maron 1985; Baron et al. 2007), though a satisfactory explanation is lacking.

We speculate that firstly, it is because of the lack of understanding that $P(t|R)$, an important part of probabilistic retrieval models, measures term recall. Secondly, the way people debug retrieval models and interact with retrieval systems may have made the mismatch problem a lot more difficult to surface than does the precision problem.

We elaborate on the second point. Given a retrieval engine, people often look at top ranked retrieval results of a given query to suggest ways to improve retrieval. When they focus on the top ranked irrelevant results, it is very easy for them to recognize the precision problem, where a certain aspect of the query semantics is missing from the top ranked results and needs to be enforced. For example, when querying for "prognosis/viability of political third parties", some top ranked false positives are about "prognosis of the next year". Given these false positives, it is very straightforward for the searcher to think that "prognosis" should be about "political third party", instead of about anything else. Thus, to remove these false positives, people will likely propose to require "prognosis" to appear close to "political third parties" in a result document, increasing matching precision. However, the real problem here is term recall or mismatch, not precision. People often do not realize this perhaps because it is not very intuitive to think that these false positives are there because "prognosis" and "viability" have been given too much emphasis by the retrieval model, suppressing a lot of the true relevant documents that mismatch these two terms, and in turn, causing the false positives that happen to match the two terms to be ranked at the top.

Instead of asking why the relevant documents are ranked so low, people tend to focus on the top ranked irrelevant results and ask why they are ranked highly. Perhaps this is why it is often difficult to realize that the real problem is often the mismatch problem, not the precision problem. Given a query with a serious mismatch problem, further restricting the match using proximity or phrases can only increase mismatch and make the situation worse.

We point out that the retrieval models and many of the queries are suffering from the mismatch problem, more than the precision problem. We observe that mismatch is a very common and very likely cause of search failure, and that mismatch is often the underlying cause of a seemingly precision related problem. This observation is very crucial, because a wrong diagnosis of the underlying causes often leads to detrimental solutions.

## 3.7 Summary

The mismatch problem is one of the most significant problems in retrieval modeling. Theoretically, for optimal term weighting in the Binary Independence Model style (which includes Okapi BM25), term recall is the only probability that is unaccounted for. Practically, term mismatch leads to the emphasis and the mismatch problems, which account for over 90% of the failures made by common IR models and techniques from the 7 state-of-the-art ad hoc retrieval systems that participated in the 2003 RIA

workshop. The precision problem only accounts for less than 10% of the failures, perhaps because these TREC queries are fairly long, and the standard retrieval models that rank highly the documents that contain many of the rare (high inverse document frequency) terms are already returning a fairly precise set of top results.

This chapter presents a set of exploratory data analyses investigating the distribution of the term mismatch probability across different terms. Results indicate that term mismatch is a probability with large variation across terms, demanding modeling and prediction. In order to promote modeling and prediction of the term mismatch probability and to inform the design of effective features, this chapter also identifies and summarizes 4 common causes of term mismatch. This chapter also explains the mechanism of how term mismatch affects retrieval performance, which suggests to use term weighting to solve the emphasis problem and CNF style Boolean expansion queries to solve the mismatch problem.

The chapter below studies the variation of the $P(t|R)$ probability for the same term across different queries.

# Chapter 4

# The Query Dependent Variation of $P(t|R)$ & its Causes

The chapter above studies the overall $P(t|R)$ variations across different terms. The basic understanding of the causes of query term mismatch guides the design of methods to predict $P(t|R)$. This chapter studies the more nuanced subject of the query dependent variation of $P(t|R)$ for the same term across different queries, with the goal of identifying ways to predict $P(t|R)$ variation for the same term appearing in different queries, which is a related but very different goal from that of the chapter above and results in different $P(t|R)$ prediction methods. This chapter focuses on data analyses, while prediction methods are examined in Chapter 5 below this.

The probability $P(t|R)$ is by definition query dependent, because $R$ is the set of documents relevant to the given query. Examples in Table 3.3 also confirm that the same term appearing in different queries may have very different $P(t|R)$ probabilities. These suggest that perhaps query dependent features are necessary to capture the query dependent variation of $P(t|R)$ and to effectively predict $P(t|R)$.

In order to guide the effective and efficient prediction of $P(t|R)$, a better understanding of the degree and the causes of this query dependent variation of the $P(t|R)$ probability is necessary. Exploratory data analyses in this chapter are carried out on 250 queries from 5 TREC datasets. They show that $P(t|R)$ does vary for the same term occurring in different queries, and query dependent features such as the level of association of the term with the query are necessary for effective $P(t|R)$ prediction. Since prior ad hoc retrieval research that predicted $P(t|R)$ only used a query independent feature – idf – for prediction, this new finding motivates the use of new query dependent features for $P(t|R)$ prediction. Analyses also show that many repeating term occurrences share very similar $P(t|R)$ values across the different queries that term $t$ appeared in. This observation can lead to more accurate or efficient $P(t|R)$ prediction methods that are based on historic $P(t|R)$ values from the training set and a prediction method that would reliably estimate the $P(t|R)$ difference between a test and a historic occurrence, which are discussed in Chapter 7.

## 4.1 Study 1 – Global Analyses of Variation

Term mismatch probability is by definition query dependent. The same term appearing in different queries can have different mismatch probabilities. Prior ad hoc retrieval research provided few clues about how to estimate $P(t|R)$ query dependently and used idf as the only feature to predict $P(t|R)$. This section investigates the variation of the $P(t|R)$ probability for the same term occurring in different queries, both to study the query dependent variation of $P(t|R)$ and to provide ideas to predict $P(t|R)$ in a query dependent way.

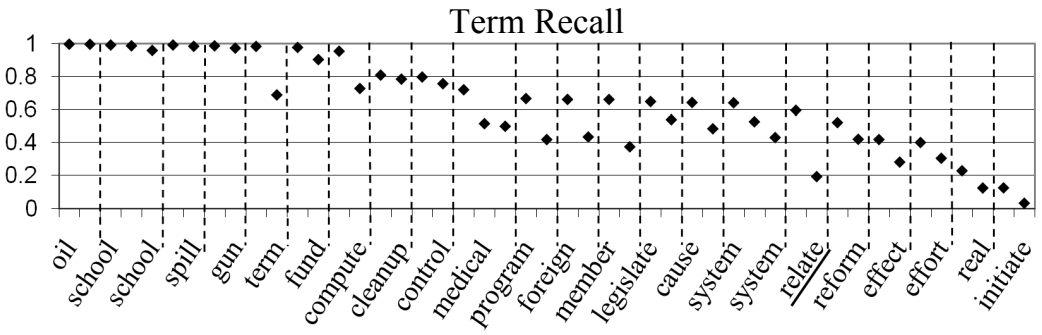



Figure 4.1: TREC 3 grouped recall probabilities of the same term recurring in different queries. The term *relate* has the largest difference across its two occurrences, with 0.1935 in query 172 "The Effectiveness of Medical Products and *Related* Programs Utilized in the Cessation of Smoking" and 0.5957 in query 183 "Asbestos *Related* Lawsuits".

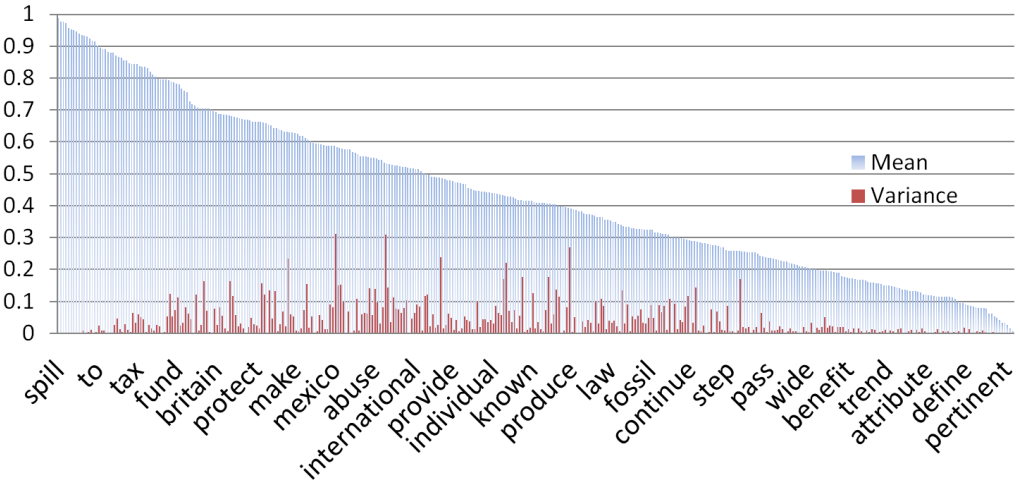



Figure 4.2: The mean and variance of term recall probabilities for the recurring description query terms from TREC 3 to 7.

In the 50 TREC 3 title queries, 22 unique terms appear in more than one query, accumulating 47 individual occurrences whose true recall values are plotted in Figure 4.1. The values for a single term are grouped within vertical dashed lines. For example, the word "medical" appeared in three queries. $P(t|R)$ for an individual term can vary from one query to another. The variation is less than 0.2 for 64% of the repeating term occurrences, occurrences where the term has already been observed in a previous query. The other 36% repeating occurrences have larger variation. This could mean that the term recall probability is more or less invariant for most term recurrences, and might help predict $P(t|R)$. However, because the set of the repeating terms from TREC 3 alone is quite small, we need to observe more to draw conclusions.

Figure 4.2 shows the means and variances of the $P(t|R)$ probabilities for the query terms that occurred more than once in the 250 queries from the Ad hoc tracks of TREC 3 to 7. Here, only terms in the description fields of the TREC queries are considered. Only prepositions are treated as stopwords and excluded. Of the 250 queries and total 1062 unique terms, 364 unique terms occur more than once, which

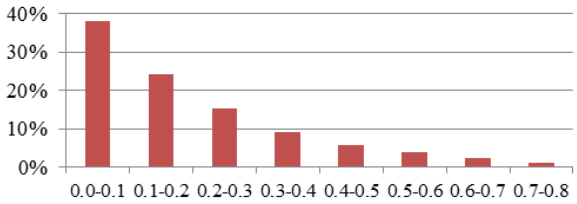



Figure 4.3: Distribution of term recall differences for the recurring description query terms on TREC 3 to 7.

is 34% of all the query terms. These 364 terms contribute to 982 repeating occurrences. Of those repeating occurrences, 62% have a term mismatch difference smaller than 0.2, and the other 38% have a difference larger than 0.2. (These percentages are accurate to $\pm$ 3% at a 95% confidence interval by the Agresti-Coull approximation, assuming future $P(t|R)$ variations are drawn from the same distribution.) The distribution of the magnitude of the $P(t|R)$ differences is shown in Figure 4.3. If $P(t|R)$ were uniformly random, only 19% of the repeats would have a smaller than 0.1 difference. Our evidence rejects the hypothesis that $P(t|R)$ for repeating terms is uniformly distributed.

In summary, these results confirm the query dependent nature of the term mismatch probability. Query dependent features are needed for predicting term mismatch. However, for most of the repeating term occurrences, the term mismatch probability does not vary much across queries. This observation suggests that for a previously observed term, we may use the previous occurrences of the term to predict the mismatch probability of the current occurrence, which could make the $P(t|R)$ prediction problem easier and the predictions more accurate.

Another observation is that the recurring query terms have an almost uniform term mismatch distribution, varying widely from 0 to 1, similar to that of all the query terms (Section 3.4). This indicates that the recurring query terms are not an easier subset of terms for mismatch prediction. In other words, if the prediction algorithm does not use any information from the historical occurrences of the term, $P(t|R)$ prediction for the recurring terms should not be much better than that for all the query terms.

## 4.2 Study 2 – A Case Study of the Extremes

To get a sense of what are the common causes of $P(t|R)$ variation, we examine some typical cases of terms having a highly different $P(t|R)$ probability across different queries.

Table 4.1 presents 6 representative cases of the 8 terms that have the largest $P(t|R)$ difference in the TREC 3 to TREC 7 description queries. The difference between two occurrences is in the range of 0.7 to 0.8.

In the first two queries, the word stem "bear" is ambiguous, sharing very *different senses*, and $P(t|R)$ also varies wildly.

The second term "disorder" is a rather abstract term. Although both queries use the same sense of the term, one query uses the term in a proper noun or fixed phrase, while the other is just a generic use of the word. Thus, *different uses* of a term can result in different $P(t|R)$ probabilities.

The 6th query "... International Convention ..." contains both a *different sense* and a *different semantic use* of the word "convention". Although "convention" is part of a fixed phrase, which is an indicator that the term might have a high recall ($P(t|R)$), the query is really just using the passing of the "convention" as a time constraint. As long as a document contains a relevant event that happened after this time point, it is considered relevant, even if the "convention" does not appear in the document, lowering $P(t|R)$.

Table 4.1: $P(t|R)$ probability variation case study. Queries and judgments are from TREC 3 to 7 Ad hoc track datasets.

| **Query with term $t$ highlighted** | $P(t\|R)$ |
|---|---|
| The frequency of vicious black **bear** attacks worldwide and the possible causes for this savage behavior | 0.9286 |
| What steps have been taken world-wide by those **bearing** the cost of E-mail to prevent excesses | 0.1429 |
| Seasonal affective **disorder** syndrome (SADS) | 0.8333 |
| Is the fear of open or public places (Agoraphobia) a widespread **disorder** | 0.1429 |
| European **Conventional** Arms Cut as it relates to the dismantling of Europe's arsenal | 0.8912 |
| International boundary disputes relevant to the 200-mile special economic zones or 12-mile territorial waters subsequent to the passing of the "International **Convention** on the Law of the Sea" | 0.1724 |
| **Mexico** City has the worst air pollution in the world. What specific steps have Mexican authorities taken to combat this deplorable situation | 0.9808 |
| How has the volume of U.S. imports of Japanese autos compared with export of U.S. autos to Canada and **Mexico** | 0.1926 |
| What are the different techniques used to **create** self induced hypnosis | 0.7143 |
| Commercial overfishing **creates** food fish deficit | 0.1192 |
| Status of trade balance with Japan - **deficit** problem | 0.7714 |
| Commercial overfishing creates food fish **deficit** | 0.0259 |

*Disjunctions in the query semantics* have a large impact on $P(t|R)$. In the 8th query about "auto imports and exports", although "Canada and Mexico" are connected with the word "and", it is in fact a disjunction. A document does not need to contain both "Canada" and "Mexico" to be relevant: one country is enough. In fact, *when assessing relevance for TREC queries, in most cases, the word "and", when serving as a logical connector, is interpreted as a disjunction*. This is likely due to the effort to make TREC assessments exhaustive, to not miss any piece of relevant information. Thus the large drop in $P(t|R)$ for the term "Mexico" is caused by the *disjunctive query semantics*.

The terms "create" and "deficit" in the last four queries did not change their word senses or word uses. However, there are still large differences in the $P(t|R)$ probabilities. We attribute this difference to the variation of a term's degrees of association with the queries. This *association variation* could be due to how people tend to describe different subjects differently, or due to the fact that different communities of people use words differently. For the word "create", "creating hypnosis" seems to be a more common combination than "creating fish deficits". According to collection statistics, $P(\text{create}|\text{hypnosis})$ is twice as much as $P(\text{create}|\text{deficit})$. For the word "deficit", governmental and financial people say "deficit" (e.g. "trade deficit") more frequently than fisherman do. Thus, a term that does not change its sense or use can still have very different frequencies of occurrence in the relevant document sets of the different queries.

Overall, of the top 8 pairs of occurrences that have the largest $P(t|R)$ difference, 3 of them are because of phrases, 2 of them are because of association variation, 2 because of multiple senses, and 1 because of disjunctive query semantics. This is a small set of cases, and this set of causes of $P(t|R)$ variation is not meant to be complete. There might be other cases of association variation or other cases of different word uses, which we have not observed in our data analysis of about 30 variation cases. However, this analysis still provides important insight into what might cause the query dependent variation of the $P(t|R)$ probability.

## 4.3 Study 3 – Medium to Low Variation Cases

The section above shows that phrases, association variation, word sense difference and disjunctive query semantics can cause extreme $P(t|R)$ variation. But, these extremes only make up less than 10% of all the term repeat cases. This section looks at some medium to low variation cases where *association variation seems to be a very common cause*.

In this study, we look at pairs of term repeats on the TREC 3 dataset. Because it is a much smaller dataset, the top 14 largest $P(t|R)$ differences range from 0.2 to 0.43, which are much smaller than the ones in Section 4.2. Of these 14 pairs, 10 of them are because of association variation, 3 of them have disjunctive query semantics, and only 1 of them is because of multiple senses.

Table 4.2: Cases of association variation causing $P(t|R)$ variation. Queries and judgments are from TREC 3 dataset.

| Query with term $t$ highlighted | $P(t\|R)$ |
|---|---|
| Ineffectiveness of **US** embargoes/sanctions | 0.8971 |
| Reform of the **US** welfare system | 0.4677 |
| Term limitations for members of the **US** congress | 0.8305 |
| Right wing Christian fundamentalism in **US** | 0.5000 |
| Impact of **foreign** textile imports on US textile industry | 0.7258 |
| US restaurants in **foreign** lands | 0.4427 |
| Impact of foreign textile imports on **US** textile industry | 0.8629 |
| School Choice Voucher System and its effects upon the entire **US** educational program | 0.5812 |
| Instances of fraud involving the **use** of a computer | 0.7881 |
| **Use** of mutual funds in an individual's retirement strategy | 0.5116 |
| Instances of fraud involving the use of a **computer** | 0.9735 |
| Stock market perturbations attributable to **computer** initiated trading | 0.7278 |

We list some of the association variation cases in Table 4.2.

We also looked at the cases where variation is even lower, 0.1 to 0.2. Of the total 7 cases, 5 are association variation, 1 disjunctive semantics, and 1 multiple senses.

Together, studies 2 and 3 show that *association variation is a very common cause of $P(t|R)$ variation. Disjunctive query semantics, word sense difference, and phrases are less frequent, but could have a large impact on the $P(t|R)$ probability*.

Studies 2 and 3 are small scale case studies, but can still inform our design of efficient $P(t|R)$ prediction methods. For example, to model association variation, tests of correlation of a query term with the query seems necessary. Term frequency in pseudo-relevant documents (top ranked documents returned from the original query) can serve as such a measure. *It is likely because of the prevalence of association variation that pseudo relevance feedback methods such as Rocchio (Rocchio 1971) or Relevance Model (Lavrenko and Croft 2001) provide very effective term weights, and also some of the local LSI features in Chapter 5 turn out to have a strong positive correlation with $P(t|R)$*. However, these features typically require an extra retrieval step, and can be expensive to compute.

## 4.4 Summary

This chapter analyzes the variation of the probability $P(t|R)$ for the same term across different queries.

Exploratory data analyses from 5 TREC datasets each with 50 queries show that term mismatch is query dependent, suggesting the use of query dependent methods for predicting the probability. At the same time, for repeating terms, the probability does not vary more than 0.2 in 62% $\pm$3% of the cases, suggesting the use of historic occurrences of a term to predict the mismatch probability of a repeating occurrence of the same term in a new query. In terms of the causes of the variation, on one hand, examining the extremely high variation cases revealed that differences in word sense or word use can cause large variations in $P(t|R)$. On the other hand, association with the query is a very common cause of variation overall.

Together with the chapter above, both chapters analyze the term mismatch probability, its query dependent variation and the causes of term mismatch and the variation. All of these are used to inform effective and efficient $P(t|R)$ prediction methods designed and tested in the Chapters 5 and 7 below.

# Chapter 5

# Predicting $P(t|R)$ – 2-pass Prediction

The previous two chapters focus on exploratory data analyses. They show that the mismatch between query terms and relevant documents happens pervasively throughout standard ad hoc retrieval collections for verbose and short queries, with a mismatch rate of 30-50% for an average query term. Given that mismatch affects retrieval accuracy and that it is an integral part of common probabilistic retrieval models, it is worth the effort to try to predict this probability. This chapter and Chapter 7 focus on predicting $P(t|R)$ effectively and efficiently.

## 5.1 Introduction

True term mismatch probabilities are calculated from relevance judgments of a given query, following directly from the definition of the term mismatch probability (see Section 3.3). However, in order to be useful for ad hoc retrieval, i.e. for a new query without any relevance judgments beforehand, a predictive approach is needed. This chapter formulates the prediction of the term mismatch probability as a standard machine learning problem. Firstly, gold standard labels (true term mismatch probabilities) are obtained from training data with known relevance judgments. Secondly, features are designed according to the factors that affect term mismatch identified in Section 3.5. Thirdly, a standard machine learning algorithm trains a statistical model on the training data and applies it to the test set. Empirical evidence is presented, showing that the features correlate well with the prediction target, and improvements in prediction accuracy are observed by using the set of effective features.

Overall, this chapter and the chapter below treat term mismatch prediction as a distinct problem, which lays the foundation for the subsequent research on applying the predicted term mismatch probabilities to improve ad hoc retrieval. End-to-end retrieval experiments are not presented in this chapter, but are presented in Chapters 6 and 8.

We argue that this intermediate step of $P(t|R)$ prediction is needed, instead of the more holistic approach of directly optimizing retrieval performance (e.g. by throwing all the features that effectively predict term mismatch into a learning to rank model). There are three main reasons for this choice of separating the retrieval task into two components of $P(t|R)$ prediction and retrieval. Firstly, the recall probability is an integral component of the retrieval models (details in Section 2.2 and 6.1), thus predicting it first, then plugging it into the retrieval models is straightforward. Secondly, separating the overall retrieval system into the two smaller and relatively independent components makes reasoning easy, and consequently simplifies experimentation, diagnosis and testing. In fact, this chapter can be seen as a unit test of the recall prediction component, while Chapters 6 and 8 are end-to-end tests. Thirdly, the term mismatch probability is a fundamental probability that is not only useful for probabilistic ranking, but

also for other retrieval techniques, such as query expansion and structured querying. More specifically, estimates of individual query term recall probabilities will not only allow us to adjust term weighting for the query terms (through automatic prediction methods), but also guide per term expansion or other interventions to fix the problem terms that need the intervention most. We are not arguing against the possibility that a holistic approach that skips $P(t|R)$ prediction could be better performing. It simply is not the focus of this thesis. In fact, the features or overall prediction designed in this work could be used in a more elaborate learning to rank model.

The prediction approach and the predictive features designed in this chapter require an initial retrieval and information from the top ranked documents to provide term weights for a final retrieval. It is called 2-pass prediction, because overall there are two retrieval steps. Chapter 7 proposes efficient 1-pass prediction methods that can be used in real-time response scenarios.

## 5.2 Modeling and Predicting Term Mismatch

The goal is to predict the recall of each term in a query without using relevance judgments for that query. This section presents a framework for using term- and query-dependent statistics to predict recall. It continues the discussion of factors that affect recall from Section 3.5, designs related features to capture these factors, and also explains the prediction/regression model.

### 5.2.1 Problem Formulation

We cast the recall prediction problem into a standard regression prediction problem, where each query term together with its estimated true recall value is treated as a sample, for training or testing. The query term is represented as a set of features, so that recall values of query terms in the training set can be generalized to predict recall for test terms previously unseen. The objective is to minimize the prediction error of term recall values.

More formally, for a set of queries $Q$ together with corresponding document collections, a training sample consists of a query term $q_i$ (from query $q$) and its true recall $P(q_i|R)$ estimated from the set of judged relevant documents $R$ for the query $q$ (as described in Section 3.3). Each term is represented as a set of features $f_{j=1..k}(q_i, q)$ depending on the corpus, the term $q_i$ and the query $q$. A regression model $M$ predicts recall as a function of the features: [1]

$$\hat{P}(q_i|R) = M(f_1(q_i, q), ..., f_k(q_i, q)) \tag{5.1}$$

Prediction error $L_Q(M)$ over the set of queries $Q$ can be measured in average L1 loss as

$$L_Q(M) = \frac{\sum_{q \in Q} \sum_{q_i \in q} |\hat{P}(q_i|R) - P(q_i|R)|}{\sum_{q \in Q} |q|} \tag{5.2}$$

where $|q|$ is the total number of terms used in the query $q$.

Here, using the recall probability to measure prediction loss is intuitive, but also an arbitrary choice. One may train over odds probability (or log odds), and then translate the predicted odds back into probability. Odds probabilities map $[0, 1)$ into $[0, +inf)$, stressing the higher probability region. In our retrieval experiments, training in odds probability is slightly more stable than in probability or log odds.

[1] In this document, $\hat{P}(t|R)$ denotes the predicted probability, while $P(t|R)$ refers to the true term recall probability, usually estimated from judged relevant documents for the query $q$.

### 5.2.2 Features – Term Occurrence

One of the main contributions of this work is pointing out some of the factors that affect term recall and designing features that represent or model them. The factors include 1) term centrality, 2) concept centrality, 3) replaceability, 4) abstractness and 5) rareness (Section 3.5).

This section develops a set of features designed to capture factors 1, 2, 3 and 5. It is not clear how to represent term abstractness, so we resort to using a feature that correlates with factor 4.

In order to compute whether a term is likely to be replaced by its synonyms or whether the concept represented by the term is important for the query, we need to know the set of synonyms for a given query term. This set of synonyms will not only depend on the query term itself, but also the very query that the term appears in. This is because data analyses in Chapter 4 show that term mismatch is query dependent. We could use external resources such as WordNet or a query log, but they may not cover all the query terms, or may introduce their own biases. Using these resources for generating query dependent synonyms is itself an important research topic.

We instead generate features using minimal external information, and restrict ourselves to only the corpus and the queries. In order to obtain synonymous relations, a term-term semantic similarity measure is needed. This term similarity measure can be obtained from a lower-dimensional (latent) subspace representation of the documents formed by Singular Value Decomposition (SVD). SVD is performed to get the singular value decomposition of the term-document matrix, the singular values are sorted in descending order, and only a small set of the largest singular values are kept. All others are set to 0. Term similarities are computed in this latent concept space.

Synonyms may not co-occur in the same document, but they typically occur in similar contexts (i.e. a higher order co-occurrence). As Kontostathis and Pottenger (2002) pointed out, Singular Value Decomposition (SVD) identifies these higher order co-occurrences (i.e. one word appearing in documents that share common words with documents containing the other word, or roughly speaking, words that occur in similar contexts, in contrast to first order co-occurrences where words appear in many common documents). As input to SVD, entries of the term-document matrix are tf*idf weighted. The specific form of $\log(tf + 1) \cdot \log((|C| + 1)/(df + 0.5))$ is used, where $|C|$ is the total number of documents in the collection.

Because the focus of this chapter is to investigate the use of term recall probabilities in term weighting, SVD is only used to provide features for predicting term recall. No expansion terms are used in the final queries. The use of expansion terms is examined in Chapter 8.

SVD can be applied globally to a corpus, or locally to top-ranked documents returned for a given query. Local SVD is more efficient and may improve the quality of synonyms, because a local SVD focuses on the senses of the terms that are actually being queried. For words with multiple senses, mutual disambiguation among query terms causes the top-ranked documents to be about the senses intended by the user, thus the synonyms identified are not about an arbitrary sense of a query term, but the sense being queried. Hull (1994) and Schütze et al. (1995) used a similar approach to generate effective document representations for a routing task, and they called the approach local LSI. This local LSI also gives a query dependent latent semantic space, which generates query dependent features for predicting term mismatch which is a query dependent probability (Section 4.1).

Denote $S(t, w_1)$ to be the inner product (similarity) of terms $t$ and $w_1$ in the latent concept space. Since SVD decomposes a term-document matrix into a term-concept matrix, a diagonal matrix of the eigenvalues, and a concept-document matrix, $S(t, w_1)$ takes the inner product of the two rows from the term-concept matrix that corresponds to the two terms. For simplicity, we also assume that terms $w_i$ are sorted in descending order of similarity to the query term $t$, i.e. $S(t, w_1) \geq S(t, w_2) \geq ...$, so that, higher in ranking, more likely synonym.

Table 5.1: Query terms and their top 5 similar terms using SVD. Queries are sampled from TREC 3 ad hoc track. 180 top ranked documents for each query are fed to SVD and the 150 largest dimensions are kept. These parameters are tuned through cross validation on the test set (TREC 4).

| Oil *spills* | | Insurance coverage which pays for long *term* care | | *Term* limitations for US Congress members | | Vitamin the cure of or cause for human *ailments* | |
|---|---|---|---|---|---|---|---|
| oil | | term | | term | | ail | |
| spill | 0.5828 | term | 0.3310 | term | 0.3339 | ail | 0.4415 |
| oil | 0.4210 | long | 0.2173 | limit | 0.1696 | health | 0.0825 |
| tank | 0.0986 | nurse | 0.2114 | ballot | 0.1115 | disease | 0.0720 |
| crude | 0.0972 | care | 0.1694 | elect | 0.1042 | basler | 0.0718 |
| water | 0.0830 | home | 0.1268 | care | 0.0997 | dr | 0.0695 |

To get an intuitive sense of what the most similar terms look like, we present in Table 5.1 a set of query terms and their most similar terms in the top ranked documents from a baseline keyword retrieval run.

It's easy to see from Table 5.1 that searchonyms extracted with SVD are query dependent. The same term "term" results in two different sets of searchonyms. But, mutual disambiguation in local SVD is not perfect. For "term" in the "term limitation" query, the word "care" which co-occurs with the "long term care" sense of "term" shows up as similar (0.0997 similarity) to the query term "term". The extracted searchonyms and similarities are not perfect. For example, one query term may become another query term's searchonym, ("spill" is identified as a searchonym for "oil"), simply because they co-occur. External resources, such as a thesauri or Wikipedia, or better searchonym extraction techniques may be used to improve these features. Nevertheless, our experiments below show that features based on SVD term similarities can be used to effectively predict recall and improve retrieval.

Overall, the SVD features depend on 2 meta-parameters, 1) the number of top retrieved documents for SVD and 2) the number of latent dimensions to keep. These parameters are tuned on a development set in experiments using 5-fold cross validation. The number of top rank documents to use in SVD increases from 180 to 600 as collection size increases, while the number of latent dimensions to keep stays around 150 for all collections. Details are presented in Section 6.2.4.1 together with the retrieval experiments.

#### 5.2.2.1 Term Centrality

The term centrality feature (Zhao and Callan 2010) measures how central a term $q_i$ is to the topic of the query, and is defined as

$$f_1(q_i, q) = S(q_i, w_{i1}) / \sqrt{S(q_i, q_i)} \tag{5.3}$$

$S(a, b)$ denotes the similarity of two terms $a$ and $b$ in terms of inner product in the latent semantic space. Term $w_{i1}$ with maximal similarity to $q_i$ is usually $q_i$ itself, thus this feature measures the length (L2-norm) of the term vector for $q_i$ in the dimension reduced concept space, and indicates how much weight of the term is preserved after dimension reduction. The maximum similarity will not be exactly 1, as the lengths of the term vectors are not normalized to 1. (When $q_i$ does not equal $w_{i1}$, we still use $S(q_i, w_{i1})$ as the feature value. We did not try to vary this strategy in experiments, as this only affects a very small number of terms, and when affected, the centrality values will not change much.) This residual weight is a measure of how close the term is to the space spanned by the top documents. For example, if a term

appears frequently in many top documents, then most of its weight will be kept. Tf.idf weighting is used for the input term-document matrix for SVD to prevent stopwords from having the highest centrality.

#### 5.2.2.2 Concept Centrality

A concept centrality feature is defined as the average similarity of the query term with its top $c$ most similar terms starting from the 2nd most similar term in the concept space.

$$f_2(q_i, q) = \sum_{i=2}^{c+1} S(q_i, w_i) / \sqrt{S(q_i, q_i)} / c \tag{5.4}$$

As in the term centrality feature, we did not try to exclude the original query term from the $c$ terms, if that term happens to be between rank 2 and $c + 1$.

Similarity in the concept space indicates the likelihood that two terms are synonyms. By taking an average of the next $c$ highest similarities, this feature prefers query terms that have many synonyms, and at the same time those synonyms have to align well with the query's latent space. Since the length of the term vector in the latent space measures term centrality, intuitively, this feature will prefer query terms that have many topically central synonyms. We fixed $c$ to be 5 from a pilot study. As evident in Table 5.1, this feature will not only capture synonyms, but also antonyms, hyponyms or even misspellings that "must be considered equivalent (to the query term) for search purposes", which Hayden (Lawlor 1962) calls searchonyms.

This feature captures the centrality of the concept represented by the original query term and its top synonyms. If a term that is similar to the original query term appears quite often in top documents, it is a good indication that the concept is necessary, even though the original query term might not be as necessary. This feature has a positive correlation with term recall.

#### 5.2.2.3 Replaceability

To measure more directly how likely the synonyms or searchonyms are to substitute for the original query term in collection documents, a replaceability feature is defined as follows (Zhao and Callan 2010)

$$f_3(q_i, q) = \sum_{j=1..6}^{w_j \neq q_i} \frac{df_j - C(q_i, w_j)}{df_j} \times \frac{S(q_i, w_j)}{S(q_i, q_i)} \tag{5.5}$$

where $C(q_i, w_j)$ is the number of documents in the collection matching both $q_i$ and $w_j$, and $df_j$ is the document frequency of $w_j$. This is a modified version of the concept centrality feature and measures how likely the original query term is to be replaced by its searchonyms in the documents. $\frac{df_j - C(q_i, w_j)}{df_j} = P(\bar{q}_i | w_j)$, and measures the likelihood that searchonym $w_j$ matches additional documents that $q_i$ does not match. Normalizing by $S(q_i, q_i)$ removes the effect of the term centrality of term $q_i$. Overall, this feature has a negative correlation (about -0.2) with recall, meaning that a more replaceable term tends to have lower term recall. Lu et al. (2009) used a very similar measure called synonym novelty to measure how many new documents a synonym can match.

#### 5.2.2.4 Abstractness

The modified terms in TREC queries are usually abstract. For example, in the query "US educational system", system is the head noun being modified by the other two terms, and it is more abstract. Since the

head is an internal node in a dependency parse tree, while the modifiers are leaves, a binary abstractness feature can be defined as whether a term is a leaf node in the dependency tree of the query. We used version 1.6.1 of the Stanford parser[2], with the output format "typedDependenciesCollapsed", so that the output can be conveniently transformed into a dependency tree. Our pilot study on the training queries shows a 0.13 correlation with recall, meaning that a more concrete term tends to have a higher recall.

The use of dependency relations to estimate term abstractness or specificity is not new. For example Caraballo and Charniak (1999) used corpus statistics of how frequently a noun gets modified by different modifiers to estimate term specificity and reported a quite promising 80% accuracy. The use of dependency parsing in IR is not new either (Metzler et al. 1984; Park and Croft 2010). The specific use of dependency parsing of queries to determine query term abstractness is a new way of applying dependency parsing in IR. In prior research, Metzler et al. (1984) focused instead on using grammatical structure to improve retrieval precision, and Park and Croft (2010) used dependency parse structure to generate features for predicting key concepts.

#### 5.2.2.5 Idf

A term rareness feature is simply defined as the inverse document frequency (idf) of a term. It was often used in prior research e.g. (Greiff 1998; Metzler 2008) to estimate term weights. Idf is a real-valued term-specific statistic. We use the specific form of $idf(q_i) = log((N - df)/df)$, where $df$ is the document frequency of $q_i$ in a corpus of $N$ documents.

#### 5.2.2.6 Clarity

Several other features have been tested, but do not perform as well as the combination of the 5 above listed features.

Term clarity is calculated as the query clarity (Cronen-Townsend et al. 2002) where the query is only consisted of the term. Term clarity is calculated by running the term as a query and computing the KL-divergence between the language model built on the several top ranked documents and the collection language model. The main idea is that the more different the results returned by the query are from the collection language model, the more salient or clear the query is.

Term clarity is a characteristic of the term itself, and initially we thought that it would correlate well with recall. Intuitively, terms with low clarity, such as 'effect' or 'deficit' will have low recall. It does perform reasonably well without the SVD features, but when combined with the SVD features, clarity hurts performance. This is perhaps because terms with low clarity would not appear consistently in top-ranked documents, thus term centrality would be low, making clarity more or less redundant.

In the experiments, we still list the retrieval performance using clarity based recall prediction as one of the baselines. The clarity scores are computed using the clarity application from the Lemur toolkit version 4.10.

#### 5.2.2.7 The Relevance Model Weight

As explained in related work (Section 2.2.6), the Relevance Model term weights can be seen as an unsupervised estimate of term recall $P(q_i|R)$. Thus, the Relevance Model weight can also be used as a query dependent feature for term recall prediction. This feature in fact has the highest correlation with recall, and when combined with the first 5 features, it provides a 5% gain in retrieval performance, although it also slightly decreases stability.

[2] http://nlp.stanford.edu/software/lex-parser.shtml

The relevance model term probability slightly differs from the term recall probability in that it estimates the multinomial term distribution over the relevant set $P_m(q_i|R)$, instead of the Bernoulli $P(q_i|R)$ probability. Thus $P_m(q_i|R)$ can be normalized to be closer to the $P(q_i|R)$ probability, and used as a feature to predict term recall.

$$f_{\text{RMw}}(q_i, q) = \frac{P_m(q_i|R)}{\max_{j=1..n,\ \&\ q_j \notin \text{Stopwords}} P_m(q_j|R)} \tag{5.6}$$

In this work, we normalize $P_m(q_i|R)$ in a per query manner, dividing each relevance model term probability by the maximum relevance model term probability of all the terms in the query, so that at least one query term in each query has a normalized feature value of 1 (see equation above). Another commonly used normalization scheme is to divide each feature value by the sum of all the feature values in a given query, instead of dividing by the maximum value:

$$f_{\text{RMw}}(q_i, q) = \frac{P_m(q_i|R)}{\sum_{j=1..n,\ \&\ q_j \notin \text{Stopwords}} P_m(q_j|R)} \tag{5.7}$$

Different normalization strategies will not affect the baseline performance of the unsupervised Relevance Model runs, and will only affect the supervised prediction methods that include in a single training dataset the Relevance Model weights from a number of different queries. This is because the normalization is done per each query, thus different queries may have different normalization constants.

Because stopwords typically have very high Relevance Model weights ($P_m(t|R)$ values) which will have a dominating effect over all the other query terms, it is important that the Relevance Model weights of the stopwords do not contribute to the normalization constant. The experiments reported in earlier work (Zhao and Callan 2010) did not exclude stopwords when computing the normalization constant for a query, and retrieval performance is slightly lower than those reported in this document.

Empirically, on the datasets that are used in this document, the difference between the two normalization schemes is small. Max-normalization is on average slightly better. When using the Relevance Model weights as the only feature, max-normalization is around 3% better on two datasets (TREC 10 and 12) and around 0.5% worse on two other datasets (TREC 6 and 8) than sum-normalization. (The two schemes perform similarly on TREC 4 and 14 datasets.) The difference between the normalization schemes become slightly larger when combining the Relevance Model weight feature with the other features, with max-normalization around 3% better on TREC 4, 8, 10 and sum-normalization only better on TREC 6. There are two reasons max-normalization may be better than sum-normalization. The first reason is that the sum becomes larger as the number of query terms in a query increases, causing feature values after normalization to be smaller than those from short queries. This bias is not completely justified. Although the average term recall probability can decrease slightly as the query gets longer, there can still be query terms in long queries that have high recall probabilities. The bias caused by sum-normalization can cause problems in prediction when the training set contains a mix of short and long queries. For example, suppose a training set contains two queries, one having two query terms with raw feature values 1 and 0 and true recall probabilities 1 and 0, and the other query containing four terms with raw feature values 1, 1, 0, and 0 and true recall probabilities 1, 1, 0 and 0. Normalizing by the maximum leads to a simple training set of 6 training instances for which target values can be perfectly predicted using the feature value directly. However, after normalizing by sum, feature values become 1, 0 for the two training terms from the first query and 0.5, 0.5, 0, 0 for the four training instances from the second query, while the prediction target values still remain the same, requiring a more complex learner than it needs be. The second reason to prefer normalizing by the maximum feature value is to be consistent with the other features. The other features are normalized to be in the range from 0 to 1, normalizing by the maximum feature value in

each query naturally gives a value distribution from 0 to 1. A more consistent normalization scheme can improve prediction performance when combining the Relevance Model weights with the other features.

Given the empirical and theoretical advantages, in the rest of the document, we only report results for the max-normalization scheme.

#### 5.2.2.8 Summary

Overall, the idf and clarity features are only dependent on the term and the corpus, not on the query. The Local LSI based features and the Relevance Model weight feature are dependent on both the term and the current query that the term occurs in.

As explained in the introduction, even though this work focuses on predicting $P(t|R)$ for query terms, broadly defined, a term does not need to be in the query, neither need to be a natural language word. Most of the features discussed here do not require so. Only the abstractness feature depends on the parsing of the query, thus requires the term to be in the query. However, it is easy to extend the definition of the abstractness feature to any collection term by doing a dependency parse of collection sentences that the term appears in, instead of the query. How effective these features will be for non query terms, or for non natural language words, is a question for future research.

### 5.2.3 Feature Preprocessing

All the feature values are first scaled linearly to be roughly from 0 to 1 before any learning happens. Instead of adapting the scaling factor to every dataset and every different meta-parameter configuration, we simply used the scaling factor of the TREC 3 training set on all the other datasets. This may affect the use of training samples from one training set to predict samples coming from a very different test set. We simply did not try to scale the features adaptively on each collection.

### 5.2.4 Prediction Model

This section describes the learning model which learns from training data the trend of how the above features can together form a prediction of the target – the term recall probability. Given the learnt model, during testing, the same features for the test sample can be fed to the learnt model to form a prediction.

Because the target of the prediction is a continuous variable, odds probability, statistical regression is typically used for learning and prediction. The prediction model takes the form of a function that maps the feature values of a given sample into the target range [0, +inf).

The rest of the section discusses what kind of a model is needed and why.

#### 5.2.4.1 Non-linearity

Features like idf do not have a linear correlation with recall (see for example Tables 3.3, 3.4), thus a non-linear regression model that can fit more complex prediction target functions (model shapes) would be a better choice. However, a non-linear model requires more training data, or a smaller number of features to avoid data sparsity. In this work, the model trains on over 400 samples and only 5 features, justifying the use of a non-linear model. More complex or more features may force the model to be linear to achieve high accuracy, such as in Regression Rank (Lease et al. 2009).

A pilot study confirms that for the current set of features, non-linear (RBF-kernel) support vector regression outperforms linear-kernel support vector regression.

Table 5.2: Pearson/Linear correlations between features and true recall, tested on TREC Ad hoc track (TREC 4, 6 and 8) and Web track (TREC 10, 12 and 14) datasets. Here, the term recall predictions $\hat{P}(t|R)$ are based on the first 5 features. The bold faced entries are the highest correlations for each dataset (on each row). The RMw column uses the Relevance Model term weights estimated by the Relevance Model RM3.

| TREC | Idf | Term Centrality | Concept Centrality | Replaceability | Abstractness | RMw | $\hat{P}(t\|R)$ |
|---|---|---|---|---|---|---|---|
| 4 | -0.1339 | 0.3719 | 0.3758 | -0.1872 | 0.1278 | 0.6296 | **0.7989** |
| 6 | -0.1154 | 0.3116 | 0.0827 | -0.3233 | 0.0963 | 0.5248 | **0.5645** |
| 8 | 0.0451 | 0.4053 | 0.2115 | -0.2892 | 0.1074 | 0.5213 | **0.6399** |
| 10 | 0.1331 | 0.3594 | 0.0899 | -0.2806 | 0.1565 | **0.5439** | 0.4425 |
| 12 | 0.1170 | 0.4767 | 0.5629 | 0.1766 | 0.1488 | 0.4596 | **0.5990** |
| 14 | 0.1972 | 0.3164 | 0.4536 | 0.0967 | 0.1024 | **0.4563** | 0.4154 |

#### 5.2.4.2 Support Vector Regression

Support vector regression shares the same idea as support vector machines. It maps the original features into a higher dimensional space, and looks for a linear solution that is as flat as possible in the higher dimension space. With a proper kernel, the trained regression model can be non-linear.

In our pilot study we tested support vector regression with linear, polynomial and RBF kernels using SVM-light version 6.02[3]. Results show that the RBF kernel performs the best. The RBF kernel measures vector similarity according to the RBF function (or the Gaussian distribution function), where the vector similarity drops exponentially with the squared distance between the vectors. Because the RBF kernel effectively localizes the impact of the training samples during testing, RBF support vector regression can fit very complex non-linear regression models.

Except simply scaling the features which happens during preprocessing, the final regression model treats all features the same. A $\gamma$ parameter controls the RBF kernel width. A larger $\gamma$ corresponds to a narrower kernel that provides even more localized effects that would fit very jagged target functions. At the same time, narrower kernel width is more likely to lead to overfitting. $\gamma$ is the third and last meta-parameter in this work. We tune the meta-parameters on development sets, and report results of 5-fold cross validation.

Another state of the art general purpose learner, boosted decision trees, was also tried. It performed slightly lower than support vector regression, but was comparable. This suggests that any reasonably powerful learner can be used on this learning problem for the set of features here designed. Results using the boosted decision tree classifier are omitted.

## 5.3 Experiments

This section examines recall prediction accuracies using different feature sets. The datasets and the preparation of the datasets are described in Section 3.2.

### 5.3.1 Feature Level Analysis

Table 5.2 shows the Pearson correlation of each feature with true recall. All of the features provide some level of predictive power. Some features correlate better with term recall e.g. Term Centrality and Concept Centrality. Some have less correlation, e.g. idf and Abstractness. All of these features provide some independent information about the prediction target, because every feature is quite different from the others.

The Relevance Model weight (RMw) is provided as a reference. It has the highest individual correlation with term recall, while supervised prediction using the first 5 features in the table usually results in higher overall correlation with term recall. Sometimes, RMw has a slightly higher correlation with term recall than the predictions based on the 5 features. This suggests that the Relevance Model weights might be effective in predicting term recall. However, this correlation is only an indirect indication of effectiveness. The end-to-end evaluation based on retrieval performance will show more clearly whether RMw is more effective than the predictions based on the 5 features.

The idf feature on more verbose queries of the earlier TREC Ad hoc track datasets (TREC 4, 6 and 8) tends to have a slight negative correlation with term recall, while it tends to have a slight positive correlation for shorter Web track queries (TREC 10, 12 and 14). Because the magnitudes of the correlations are small, random variations in the data could be causing the change. There might be other causes of such a variation. One possible reason is that the later Web track queries tend to be more succinct than the earlier Ad hoc track queries. Being more succinct tends to make the query terms to have higher recall. Another possible reason is that the later Web track datasets with much larger document collections tend to have a less complete set of judgments, causing the judgments to give more bias toward matching most of the query terms, especially those with high idf. These factors might contribute to the slight positive correlation between idf and term recall on the Web track datasets, but a negative correlation for the Ad hoc datasets.

The Abstractness and Replaceability features have a lower absolute correlation with term recall. We conjecture that firstly, abstractness may not be a frequent cause of mismatch for most of the query terms, and that secondly, the two features are perhaps not very accurate approximations of the true abstractness and replaceability variables. For example, on the larger TREC 12 and 14 document collections, the Replaceability feature changes behavior and becomes positively correlated with term recall. This is not the intended behavior of the feature. More detailed data analyses may show how accurate these features are at modeling the potential causes of term mismatch. Since the focus of this work is the overall prediction framework, we leave further improvements as future work.

### 5.3.2 Recall Prediction Accuracy

This section measures the accuracy of several term recall prediction methods in per term average absolute prediction error (L1 loss) as defined in Section 5.2.1. The TREC 3 dataset with relevance judgments is used as training data and TREC 4 as test data. We do not apply this feature selection step for the rest of the TREC test datasets to avoid overfitting. The best features selected from TREC 4 is used in term recall prediction experiments for the other datasets.

Table 5.3 shows the performance of several prediction methods, including a baseline that always predicts 0.55 (the average training set term recall), support vector regression with different sets of features, and predicting from the true recall of a previous occurrence of the same term. With idf as the only feature, prediction is slightly worse than the baseline, indicating that using idf as the only feature as done in prior research (Greiff 1998; Metzler 2008) does not predict term recall effectively. After adding abstractness

[3] http://svmlight.joachims.org

Table 5.3: Term recall prediction accuracy, training on TREC 3 titles and testing on TREC 4 descriptions. (TREC4 queries as provided by TREC do not include titles.) Lower L1 loss is better, and negative changes in L1 loss represent improvements in prediction accuracy.

| Prediction method | Avg L1 loss | Change over Baseline |
|---|---|---|
| Baseline: Average (constant) | 0.2936 | 0% |
| Idf | 0.3078 | +4.83% |
| Idf+Abstractness+Clarity | 0.2539 | -13.52% |
| Idf+Abstractness+3 SVD features 1-3 | 0.1945 | -33.75% |
| Tuning meta-parameters | 0.1400 | -52.32% |
| TREC 3 previous occurrences | 0.1341 | N/A |

and clarity features, error drops by 13%. The three SVD features clearly outperform clarity, where meta-parameters are defaulted to 1000 feedback documents, 100 latent dimensions and $\gamma$ equals 1.5. 5-fold cross validation finds the meta-parameters to be 180, 150 and 1.5 which further decreases prediction error to half of the baseline. Removing any of the 5 features hurts performance.

The last row of Table 5.3 uses the true recall of a previous occurrence of a term to predict the recall of another occurrence of the term in a different query. Different from previous methods, this was tested on recurring terms of the TREC 3 training set. Given that the recall distribution of recurring terms (Figure 4.1) is quite similar to that of all query terms, the prediction of recall on just recurring terms should not be much easier than on all terms. This low prediction error of 0.1341 shows promise in this history based recall prediction method. It also indicates that for many occurrences of the same term, because a majority sense or majority use dominates, recall varies little across those queries, and a term dependent prediction method may perform well enough for most terms. Chapter 7 examines how well history based recall prediction works.

### 5.3.3 Efficiency

This chapter is mostly concerned with accuracy, but also lists processing times for the various stages of predicting term recall. All these experiments were run on a Dell 2850 server with 2x 2.8 GHz dual-core Intel Xeon processors and 8 GB of RAM, and were run using the Indri search engine from the Lemur toolkit version 4.10, slightly adapted for the additional functionality.

Randomly accessing hundreds of document vectors from disk (to prepare the input matrix for SVD) can be expensive. On average, retrieving 200 document vectors (about 10,000 unique terms) from a smaller TREC 3-8 index takes about 0.2 seconds. For the larger GOV2 index, retrieving 600 document vectors (about 30,000 unique terms) takes about 10 seconds (wall clock time, about 1 second CPU time). This can be much faster if the document vectors are cached in memory, or grouped into topical shards (Kulkarni and Callan 2010) instead of hundreds of totally random disk accesses. Further speedup is possible using a Reverted Index (Pickens et al. 2010) that specializes in fast feedback retrieval.

SVD and dependency parsing are the next most expensive tasks. Per query, SVD on 600 documents with 30,000 terms takes about 8 seconds (wall clock and CPU time). Further speedup is possible. Since the goal of SVD is just to find possible searchonyms of query terms, SVD does not need to converge; a smaller number of iterations may suffice.

Dependency parsing takes about 1 second per query (15 words) using the Stanford parser which not only does dependency parsing but also a full syntactic constituent parsing. Faster versions of dedicated dependency parsers exist, e.g. the MST Parser (McDonald et al. 2005). Further speedup is also possible, as

the feature being used only indicates whether a term is a leaf node in the parse tree, and does not require full knowledge of the whole parse tree. Thus, simpler but more direct dependency leaf identification methods would suffice. Alternative term abstractness measures can also help avoid parsing the queries.

Recall values do not vary more than 0.2, for most (two thirds) of the occurrences of recurring terms. This regularity can be utilized to speed up recall prediction. Whenever we are certain that the recall value of a previous occurrence of the same term can be reused, we can avoid all the feature generation and prediction stages altogether. Efficient term recall prediction is the focus of Chapter 7.

## 5.4 Discussion – Transcendental Features and Retrieval Modeling as a Transfer Learning Task

This regression prediction framework for predicting $P(t|R)$ is just a simple and straightforward application of regression learning from statistics. However, a significantly new idea here is that the learning happens across the different queries, which is consistent with the transfer learning formalism.

Traditionally, the retrieval of collection documents according to a given query is treated as one document classification task, where each document needs to be classified as relevant or non-relevant to the query. Thus, a retrieval model $\phi$ maps a document $d$ in the document collection $C$ to a Boolean value (relevant or not).

$$\begin{aligned} & \phi(d) \in \{R, N\}, \forall d \in C \\ \text{or} \qquad & \phi : C \mapsto \{R, N\}. \end{aligned} \tag{5.8}$$

This view was adopted by the Binary Independence Model (Robertson and Spärck Jones 1976) as well as many subsequent models.

However, a retrieval model can be viewed more generally, if we consider the query $q$ to be a variable in the model. A retrieval model $\Phi$, given a query $q$, produces a classifier $\phi$. Thus the retrieval model is at a higher level of abstraction than the classifier.

$$\Phi(q) = \phi_q : C \mapsto \{R, N\} \tag{5.9}$$

This formalism treats each query as a classification task, as in (Robertson and Spärck Jones 1976; Anagnostopoulos et al. 2006). What's different is that now the retrieval model is responsible for all the possible queries. The retrieval model takes in a query as input and produces as output a document classifier for that specific classification task. This means the retrieval modeling task is a meta-classification task, and the retrieval model is a meta-classifier.

Given this more general view, training a learner on a set of training queries and applying the learnt model to a test query becomes a *transfer learning* problem (Do and Ng 2005). In the traditional classification setup, the collection statistics from individual terms are used as features to predict relevance scores. In the more general transfer learning setup, in order for the learnt model to be able to generalize across different queries, the features can no longer be as simple as term statistics, but instead, need to transcend individual queries (e.g. centrality replaceability etc.) and need to vary with the changing queries. In this work, for a new test query $q$, the term weights are adjusted based on the predicted $\hat{P}(t|R)$ values. This means a completely new classifier with the new feature (term) weights is created by the meta-learner and used to classify collection documents for this new query.

In the light of this general understanding, the $P(t|R)$ prediction problem becomes a meta-learning problem, and is one level of abstraction higher than the traditional understanding of retrieval as text classification. This may also be part of the reason why prior work, working at a lower abstraction level, had difficulty identifying effective features for $P(t|R)$ prediction.

The transfer learning framework has the flexibility to naturally incorporate relevance feedback information for test queries, and at the same time still takes advantage of a set of training queries. Existing single-classifier frameworks such as learning to rank only train one single classifier from the set of training queries, and cannot easily utilize relevance feedback information that may become available during testing.

## 5.5 Summary

The term mismatch probability by definition can be estimated from relevance judgments of a given query. This is useful for two scenarios, 1) getting the true term mismatch probabilities for data analyses, and 2) generating labels for training prediction models. In order to be used for ad hoc retrieval where relevance judgments are unknown for the test query, a regression based prediction approach is designed.

One important contribution of the work is the design of numeric features that try to capture the factors that affect the term mismatch probability. These features make better predictions of the term mismatch probability possible. The features designed here are shown to correlate well with the true mismatch probabilities, and overall lead to an accurate prediction of the term mismatch probability.

Overall, this work establishes a simple and general framework for analyzing relevance information, designing effective features for mismatch prediction and evaluating these features. This framework only relies on queries and their associated relevance judgments which can be found in any standard test collection for retrieval evaluation. This work has populated the framework with a set of features based on an initial set of hypotheses of what might cause term mismatch. Others can use this framework to design and test features that explore other hypotheses.

This chapter deals with effective $P(t|R)$ prediction. The chapter below specifically explores the use of the 2-pass predicted $\hat{P}(t|R)$ probabilities as term weights, to improve over traditional retrieval models such as Okapi BM25 or language models that are based on tf.idf-only term weighting.

# Chapter 6

# $P(t|R)$ Term Weighting Retrieval

The chapter above designs new features and methods to predict $P(t|R)$, the term recall probability. This chapter and Chapter 8 below aim to show how better recall predictions can bring a significant gain in ad hoc retrieval performance.

This chapter first provides theoretical justifications of how to apply term recall to different retrieval models, such as the probabilistic retrieval model BM25, and language models. It then goes on to show experiments using true and predicted term recall values to improve ad hoc retrieval performance, over several standard baselines.

Overall, this chapter provides a first application of the predicted term mismatch probabilities, and a first set of experiments that show recall prediction to be a promising direction to pursue. All of these motivate further research applying recall prediction in structured query expansion and other retrieval interventions.

## 6.1 Theory

In the related work (Section 2.2.1), it is shown that term recall is one of the two probabilities that make up the Binary Independence Model which is one of the earliest probabilistic retrieval models. This section explains in more detail how term recall probability can be instantiated in modern retrieval models, such as Okapi BM25 and statistical language models. For BM25 and BIM, the term recall probability appears in the model and the predicted recall probabilities can be directly plugged into the model, while for statistical language models which do not model term relevance, the recall probabilities are applied as user term weights in search queries.

### 6.1.1 BIM and Okapi BM25

The Okapi BM25 model (Robertson et al. 1995) extends the Binary Independence Model, utilizing the RSJ term weighting (Equation 2.2), but also considers properties such as probabilistic indexing and document length normalization, all in a simplified Two-Poisson (eliteness) model (Robertson and Walker 1994). For every term, two Poisson distributions, elite and nonelite, are used to model within-document term occurrence/frequencies. Eliteness is a hidden variable, used to capture the extent that the document is about the concept represented by the term. The BM25 model is constructed so that if a query term is fully elite for a document, it will contribute the full RSJ term weight during retrieval scoring; if less elite, the term weight will be smaller than the RSJ weight.

Because the limiting case of BM25 term weighting is just the RSJ term weight, as term frequency goes to infinity (eliteness goes to 1), adapting term recall into the BM25 model is just plugging the predicted term recall probabilities directly into the RSJ term weight (Equation 2.2).

The recall-enhanced BM25 model produces the following retrieval score:

$$Score(D,q) = \sum_{t \in q \cap D} \log \left( \frac{\hat{P}(t|R)}{1-\hat{P}(t|R)} \cdot \frac{1-P(t|\bar{R})}{P(t|\bar{R})} \right) \cdot \frac{\mathrm{C}(t,D) \cdot (k_1+1)}{\mathrm{C}(t,D) + k_1(1-b+b\frac{|D|}{avgdl})} \tag{6.1}$$

where $\hat{P}(t|R)$ is predicted term recall of $t$ for query $q$. $P(t|\bar{R})$ is approximated with $P(t|C) = \frac{df_t}{N}$, $N$ being the number of documents in the collection. $\mathrm{C}(t,D)$ is the term frequency for $t$ within document $D$. $|D|$ is the length of the document measured in the number of tokens in the document, and $avgdl$ is the collection average document length. Two free parameters are $k_1$ and $b$.

### 6.1.2 The Relevance Model

In the statistical language modeling framework, the Relevance Model (Lavrenko and Croft 2001) provides the relevance based term weighting. The original generative language model (Ponte and Croft 1998; Zhai and Lafferty 2001) directly scores a document by how likely the document language model is to generate the query.

$$Score(D,q) = \sum_{t \in q} \log P_m(t|D) \tag{6.2}$$

where $P_m(t|D)$ denotes the multinomial document language model.

The Relevance Model (Lavrenko and Croft 2001) provides a way to model relevance in the language model framework. Given a search query, a relevance model is an ideal term distribution $P_m(t|R)$ for all $t$, estimated from true relevant documents, assuming complete relevance judgments are given. When there are no relevance judgments for the current query, the relevance model can be estimated in an unsupervised manner from the top documents of an initial retrieval, thus, instantiated as a query expansion and pseudo-relevance feedback method. The final retrieval score of a result document is given by the KL-divergence between the relevance model and the document model, or equivalently, weighting each term by its relevance model probability:

$$Score(D,q) = \sum_{t \in q} P_m(t|R) \cdot \log P_m(t|D) \tag{6.3}$$

where $P_m(t|R)$ is the relevance model, a multinomial distribution over all terms in the vocabulary, thus, $\sum_{t \in V} P_m(t|R) = 1$. In contrast, for the Bernoulli term recall probability, for any $t \in V$, $P(t|R) + P(\bar{t}|R) = 1$.

#### 6.1.2.1 Modeling Assumptions for the Document Model

Under the language modeling retrieval framework, a language model estimated from one particular document is used to estimate the generation probability for the query, and to score that particular document.

Multiple Bernoulli distributions (Ponte and Croft 1998; Metzler et al. 2004; Losada 2005) multinomial distributions (Zhai and Lafferty 2001) and multiple Poisson distributions (Mei et al. 2007) have been used to estimate the document model, and shown to be comparable and successful in many retrieval scenarios.

These models typically take into account term frequency and document length information. For example, the multiple Bernoulli and the multinomial models treat the occurrence of a term in one particular

position of a document as a Bernoulli or multinomial trial. These can be called *multiple pocket models*, because the models assume multiple pockets that a term can appear in a document, with each pocket being one word occurrence in the document. Under multiple pocket models, a document $d$ has $dlen(d)$ pockets, where $dlen(d)$ is the length of the document in number of words.

#### 6.1.2.2 Modeling Assumptions for the Relevance Model

The modeling of the relevance term distribution requires additional modeling of the process of generating one particular relevant document from the set of relevant documents. For example, Lavrenko and Croft (2001) assumes that the relevance term distribution is a single multinomial term distribution estimated from several multiple pocket multinomial document models. However, the original $P(t|R)$ probability from (Robertson and Spärck Jones 1976) (which inspired the Relevance Model work (Lavrenko and Croft 2001)) uses instead multiple Bernoulli distributions to model term occurrences in relevant documents, where each Bernoulli trial represents whether the term appears in one relevant document or not. The multiple Bernoulli models in (Robertson and Spärck Jones 1976) do not distinguish different positions in a document, and can be called *single pocket models*.

The multiple pocket models and the single pocket models differ mainly in how term occurrence probabilities are estimated from a document. The multiple pocket models treat each term in a document as one trial, thus take into account term frequency and document length in its estimation. The single pocket models, on the other hand, do not take into account term frequency and document length information. Given these different types of models, it is not well understood what's the best way to estimate the relevance model.

Nevertheless, the single pocket multiple Bernoulli term distributions can be normalized into one multinomial term distribution and fitted into a relevance model, as shown below.

$$Score(D,q) = \sum_{t \in q} \frac{\hat{P}(t|R)}{\sum_{t' \in q} \hat{P}(t'|R)} \cdot \log P_m(t|D) \tag{6.4}$$

where $\hat{P}(t|R)$ is the predicted term recall.

It is possible to fit the recall probabilities into a multinomial relevance model, because first and foremost, the Relevance Model provides a representation that allows one to plug in relevance based term weights into the language model retrieval framework. Although the original Relevance Model work provided a particular way of estimating these weights (in a multiple pocket single multinomial model), other estimation methods can be used. In fact, the above formula provide one way of estimating the relevance model using recall probabilities that come from several single pocket Bernoulli models.

A further connection exists showing that after several simplifying assumptions, the model probabilities in a specialized multiple pocket multinomial relevance model can become proportional to the single pocket multiple Bernoulli estimated term recall probabilities. Following Equation 2.9,

$$\begin{aligned} P_m(t|R) = & \sum_{D \in R} P_m(t|D) \cdot P(D|R) \quad \text{(assuming } P_m(t|D,R) = P_m(t|D) \text{ )} \\ = & \sum_{D \in R} \frac{\mathrm{C}(t,D)}{|D|} \cdot P(D|R) \\ = & \sum_{D \in R} \frac{\mathrm{C}(t,D)}{|D| \cdot |R|} \quad \text{(assuming uniform document prior in the relevant class)} \end{aligned} \tag{6.5}$$

On a side note, document prior in the relevant class may not be uniform. For example, in the case of graded relevance, documents may exhibit degrees of relevance. The Relevance Model formalism can still model these cases using a non-uniform $P(D|R)$ distribution.

Further assume that term occurrences are binary, (tf is either 0 or 1), and suppose relevant documents all have the same length $L$. The relevance model probability $P_m(t|R)$ is only a constant factor off its recall:

$$\begin{aligned} P_m(t|R) = & \sum_{D \in R} \frac{\mathrm{C}(t,D)}{|D| \cdot |R|} \\ = & \frac{1}{L} \cdot \sum_{D \in R} \frac{\mathrm{C}(t,D)}{|R|} && \text{(assume relevant docs have length } L\text{)} \\ = & \frac{1}{L} \cdot \frac{\sum_{D \in R} \mathrm{C}(t,D)}{|R|} \\ = & \frac{1}{L} \cdot P(t|R) && \text{(assume binary term occurrence)} \end{aligned}$$

However, without assuming binary term occurrences and uniform document length, the relevance model probabilities will be different from the term recall probabilities, as shown empirically in Section 6.2.3.2.

Despite the connection based on simplifying assumptions, this work focuses on the supervised prediction of the recall probabilities, which provides a novel and effective way of estimating the relevance model, and can be used to improve the traditional unsupervised estimation of (Lavrenko and Croft 2001). At the same time, term recall, being independent of the average length of the relevant documents, is a more direct and easier objective for prediction than the near 0 multinomial term probabilities of the Relevance Model.

No prior work discussed whether the multiple pocket models or the single pocket models should be used for estimating the relevance model, thus, we test both assumptions empirically.

The main difference between single pocket and the multiple pocket estimations lies in whether multiple term occurrences in the same document is rewarded. Multiple pocket models reward high term frequency terms, while single pocket models are only concerned with whether the term appeared in a relevant document regardless of the number of term occurrences in the document.

There are many ways to reward multiple term occurrences. The simplest is to weight the binary term occurrences in relevant documents by their term frequencies, which we call absolute tf weighting:

$$P_m^{abs}(t|R) = \frac{\sum_{D \in R} \mathrm{C}(t,D)}{\sum_{D \in R} max(\mathrm{C}(t,D), 1)} \tag{6.6}$$

Applying this estimate as user term weight (as in Equation 2.9) leads to the "Multinomial-abs" runs in our experiments (in Table 6.1 below). This estimation approach is not effective, as the estimates are usually biased by long documents, where term frequency of the term $t$ – $\mathrm{C}(t,D)$ – can be very high, dominating both of the two summations in the numerator and denominator, resulting in a close to 1.0 estimate even when the term only appeared in a few of the relevant documents.

Thus, another way to estimate the relevance model is to normalize the term frequency by document length, which leads to the multinomial relevance model estimates (Equation 6.5):

$$P_m(t|R) = \sum_{D \in R} \frac{\mathrm{C}(t,D)}{|D| \cdot |R|}$$

The single pocket multiple Bernoulli models lead to a better retrieval performance. Section 6.2.3.2 below presents the results and provides a detailed explanation of why this may be the case. Overall, the multiple pocket document models are not theoretically consistent with the single pocket relevance models, which demands a cleaner explanation. We point out this problem and leave it as future work.

## 6.2 Experiments – Retrieval Using 2-pass $P(t|R)$ Prediction

This section presents retrieval experiments using recall based term weighting for both estimated true recall values from relevance judgments and predicted recall based on supervised learning. Ablation studies are presented to provide a sense of what features are most effective. The basic baselines are the language model with Dirichlet smoothing and Okapi BM25. However, other baselines such as Relevance Model are also included to provide a better understanding of how and why recall term weighting works.

From a pilot study, we fixed the baseline Dirichlet prior $\mu$ at 900 for TREC 3-8, 11-12, and 1500 on other sets. For BM25, $k1 = 1.2$, $b = 0.75$.

All experiments and significance tests were run using the Lemur/Indri toolkit version 4.10. Small modifications were made to allow for setting term recall weights into the RSJ weight for the BM25 baseline, and for running local SVD to obtain term similarity values.

### 6.2.1 Evaluation Metrics

Retrieval accuracies are reported in Mean Average Precision (MAP) of the top 1000 results, which is also used as the objective for tuning the meta-parameters on the development sets. For each query, given a rank list of documents (up to rank 1000) and a set of known relevant documents, Average Precision takes the average of the precision values at the ranks of every relevant document. Precision at rank K is measured as the number of relevant documents at or above rank K divided by K. Taking the arithmetic mean of the Average Precision numbers for every query in the test set results in the MAP measure. The Average Precision measures overall retrieval success throughout the rank list, and is widely used as a standard measure in evaluation forums like TREC. MAP is a preferred ranked retrieval metric due to its discrimination and stability (Manning et al. 2008).

Also reported is precision at rank K, in particular, precision at 10 (P@10) or 20 (P@20) to measure top rank precision. These measures are insensitive to any kind of result swaps within the top K or any kind of rank list changes after rank K. These measures are commonly used in Web search scenarios where top precision is important. We list the top precision measures as reference.

### 6.2.2 Significance Tests

Smucker et al. (2007) recommended the two tailed randomization test for measuring the statistical significance of a difference between mean performance measures, say MAP. The randomization test is based on the Null hypothesis that the difference between two systems (system A and system B) comes from randomly assigning the labels A and B to the results generated by the two systems. This allows the test to be applied on numeric values instead of just signed changes (improve vs decrease) used in the sign test.

The sign test exhibits very different characteristics from the randomization test, and may make false alarms when the randomization test does not, thus Smucker et al. (2007) recommended to abandon the sign test when comparing numeric values such as Mean Average Precision (MAP).

We draw different conclusions from that work (Smucker et al. 2007). The different characteristics of the two tests allow us to view test results from two different perspectives. The randomization test takes the magnitudes of the differences into account, but because of that, may favor runs with outliers far from the baseline. The sign test does not take into account the magnitude of the difference, thus is robust to outliers, but at the same time may favor runs which improve slightly on many queries, but fail wildly on a small set of the queries.

Thus, in order to avoid false alarms and draw safer conclusions, in this work, both two-tailed paired randomization test and two-tailed paired sign test were used. In the experiments in this work, the two tests

rarely disagreed with each other, and when they did disagree, an analysis and a discussion was provided.

### 6.2.3 True $P(t|R)$ Weighting

This section evaluates the case where relevance judgments are used to estimate term recall (as per Section 3.3). True recall values are used to weight query terms, and retrieval is evaluated on the same set of relevance judgments that provided the recall term weighting. This is only intended to show the potential of using recall predictions as term weights, similar to the retrospective case of Robertson and Spärck Jones (1976). To our knowledge, this is the first work to report performance from applying true recall weights on retrieval models other than BIM. Improvements over state-of-the-art models underscore the potential of applying recall prediction.

Table 6.1: Retrieval performance with *true* recall weighted query terms, in Mean Average Precision. Bold face means significant by both randomization and sign tests with significance level $p< 0.05$. Queries are generated from TREC *description* query fields.

| TREC dataset | Document collection | Topic numbers | LM *desc* Baseline | LM *desc* $P(t\|R)$ | Improvement | $p$ - randomization | $p$ - sign test |
|---|---|---|---|---|---|---|---|
| 4 | disk 2,3 | 201-250 | 0.1789 | **0.2703** | **51.09%** | 0.0000 | 0.0000 |
| 6 | disk 4,5 | 301-350 | 0.1586 | **0.2808** | **77.05%** | 0.0000 | 0.0000 |
| 8 | d4,5-cr | 401-450 | 0.1923 | **0.3057** | **58.97%** | 0.0000 | 0.0000 |
| 9 | WT10g | 451-500 | 0.2145 | **0.2774** | **29.32%** | 0.0000 | 0.0005 |
| 10 | | 501-550 | 0.1627 | **0.2271** | **39.58%** | 0.0000 | 0.0000 |
| 12 | .GOV | TD1-50 | 0.0239 | **0.0868** | **261.7%** | 0.0000 | 0.0000 |
| 14 | .GOV2 | 751-800 | 0.1789 | **0.2674** | **49.47%** | 0.0001 | 0.0002 |

Table 6.1 shows that on different datasets, using true recall term weights gives a consistent 30%-80% improvement over the baseline model on description queries. Bold faced results are significantly better than the baseline by both significance tests. Baseline is language modeling with Dirichlet smoothing. The MAP on TREC 12 dataset is small, but comparable to the results reported in the TREC proceedings. This large potential of gain in using $P(t|R)$ for term weighting suggests that a $P(t|R)$ prediction method does not need to be perfect to outperform the baseline, and the large potential also supports more work on trying to improve term recall prediction.

We note that the performance gain using true recall term weights is similar to the gain reported on the WT10g (TREC 9, 10) collection by the best sampled term weights (Lease et al. 2009). The Oracle case of regression rank (Lease et al. 2009) directly tuned term weights using MAP as objective, thus, this close performance means true recall weights are close to the best sampled term weights of Lease et al. (2009) in retrieval effectiveness.

#### 6.2.3.1 Short v.s. Long Queries

Short queries (denoted as *title* in the tables) sometimes (4 out of 6 datasets) perform better than long description queries (*desc* in tables). Applying true recall weights on title queries gives steady but smaller improvements over title baselines. True recall weighted description queries outperform true recall weighted titles simply because there are more terms to tune weights on. For the same reason, predicted recall applied to title queries does not lead to a significant change in retrieval performance (see Table 6.7). In the short query case, expansion terms are expected to have more impact.

Table 6.2: Retrieval performance (MAP) with true recall weighted query terms - short v.s. long queries. Queries generated from title fields are denoted as *title*, and those from description fields are denoted as *desc*. Bold face means significant by both randomization and sign tests with significance level $p< 0.05$, compared to the corresponding baselines. TREC 4 queries do not have the title field, thus results for title queries are not available.

| TREC dataset | LM *title* - Baseline | LM *title* - $P(t\|R)$ | Improvement | LM *desc* - Baseline | LM *desc* - $P(t\|R)$ | Improvement |
|---|---|---|---|---|---|---|
| 4 | N/A | N/A | N/A | 0.1789 | **0.2703** | 51.1% |
| 6 | 0.2362 | **0.2514** | 6.44% | 0.1586 | **0.2808** | 77.0% |
| 8 | 0.2518 | 0.2606 | 3.49% | 0.1923 | **0.3057** | 59.0% |
| 9 | 0.1890 | **0.2058** | 8.89% | 0.2145 | **0.2774** | 29.1% |
| 10 | 0.1577 | **0.2137** | 35.5% | 0.1627 | **0.2271** | 36.2% |
| 12 | 0.0964 | **0.1042** | 8.09% | 0.0239 | **0.0868** | 263% |
| 14 | 0.2511 | **0.2674** | 6.49% | 0.1789 | **0.2674** | 49.5% |

#### 6.2.3.2 Single Pocket v.s. Multiple Pocket Relevance Models

Table 6.3: Retrieval performance (MAP) with true recall weighted query terms. The single pocket multiple Bernoulli estimates lead to the Recall runs, and the multiple pocket multinomial estimates lead to the Multinomial-abs and Multinomial RM runs. Bold face means significant by both randomization and sign tests with significance level $p< 0.05$, compared to the LM baseline.

| TREC dataset | LM *desc* - Baseline | LM *desc* - $P(t\|R)$ | Multinomial-abs *desc* | Multinomial RM *desc* |
|---|---|---|---|---|
| 4 | 0.1789 | **0.2703** | **0.1988** | **0.2613** |
| 6 | 0.1586 | **0.2808** | **0.2088** | **0.2660** |
| 8 | 0.1923 | **0.3057** | **0.2345** | **0.2969** |
| 9 | 0.2145 | **0.2774** | 0.2239 | **0.2590** |
| 10 | 0.1627 | **0.2271** | 0.1653 | **0.2259** |
| 12 | 0.0239 | **0.0868** | **0.0645** | **0.1219** |
| 14 | 0.1789 | **0.2674** | **0.2150** | 0.2260 |

This section compares the single pocket Bernoulli term recall $P(t|R)$ estimates which do not consider term frequency and document length information with the multiple pocket models which takes such information into account when modeling term relevance. Two multiple pocket models were tested, the Multinomial-abs method (Equation 6.6) and the Multinomial RM method (Equation 6.5).

The Multinomial-abs method weighs the binary term occurrences in the relevant documents by their term frequencies:

$$P_m^{abs}(t|R) = \frac{\sum_{D \in R} \mathrm{C}(t, D)}{\sum_{D \in R} max(\mathrm{C}(t, D), 1)}$$

Applying this estimate as user term weight (as in Equation 2.9) leads to the "Multinomial-abs" row of Table 6.3. The title (short) queries are dropped from this table because the improvements are small and not enough to distinguish among the different methods.

This estimation approach is not very effective, as the estimates are usually biased by long documents, where tf of the term $t$ – $\mathrm{C}(t, D)$ – can be very high, dominating both of the two summations in the

numerator and denominator, resulting in a close to 1.0 estimate.

A better way to estimate the relevance model is to normalize the term frequency by document length, which is just the multinomial relevance model (Multinomial RM) estimation:

$$P_m(t|R) = \sum_{D \in R} \frac{\mathrm{C}(t, D)}{|D| \cdot |R|}$$

Applying the multinomial estimates as user term weights leads to the results in the "Multinomial RM" row in Table 6.3[1].

The Multinomial RM is quite stable but consistently and slightly worse than the single pocket term recall estimates on all test collections except on TREC 12. The improvement for Multinomial RM is not significant on TREC 14, by sign test. This less significant result might be due to the burstiness of term occurrences. Even though, document length normalization is taken care of in the "Multinomial RM", when a term $t$ occurs in a relevant document, it is more likely to occur again, resulting in overly confident estimates of $t$'s occurrence in the relevant set. Previous retrieval models typically discount term frequency either applying a log transformation ($\log tf$) or using similarly shaped $\frac{tf}{1+tf}$ transformation (Robertson and Walker 1994).

In summary, for estimating the relevance model, term frequency and document length normalization for the relevant documents is not important: The single pocket multiple Bernoulli models which do not use such information are more effective than the multiple pocket multinomial models. This is consistent with the observation by Buckley and Salton (1995), who assigned weights to expansion terms in Rocchio relevance feedback, and found that term occurrences in long relevant documents should not be down weighted according to the lengths of the relevant documents. On the other hand, when estimating the document language models, term frequency and document length normalization has always been important. A more unified model for both document model and relevance model estimation is needed. Since it is not the focus of this thesis, we leave this as future work.

#### 6.2.3.3 Language Model v.s. Okapi BM25

Table 6.4: Retrieval performance (MAP) of language model and Okapi BM25 with true recall weighted query terms. BM25 parameters are the default $k1 = 1.2$ and $b = 0.75$ for Okapi Baseline and Recall columns, and set at $k1 = 0.9$ and $b = 0.5$ for the Okapi tuned column. Bold face means significant by both randomization and sign tests with significance level $p< 0.05$.

| TREC | LM *desc* - Baseline | LM *desc* - $P(t\|R)$ | Okapi *desc* - Baseline | Okapi *desc* - $P(t\|R)$ | Okapi tuned *desc* - $P(t\|R)$ |
|---|---|---|---|---|---|
| 4 | 0.1789 | **0.2703** | 0.2055 | **0.2679** | **0.2773** |
| 6 | 0.1586 | **0.2808** | 0.1773 | **0.2786** | **0.2881** |
| 8 | 0.1923 | **0.3057** | 0.2183 | **0.2894** | **0.3032** |
| 9 | 0.2145 | **0.2774** | 0.1944 | **0.2387** | **0.2630** |
| 10 | 0.1627 | **0.2271** | 0.1591 | **0.2003** | **0.2199** |
| 12 | 0.0239 | **0.0868** | 0.0449 | **0.0776** | **0.0656** |
| 14 | 0.1789 | **0.2674** | 0.2058 | **0.2403** | **0.2631** |

With the default parameter values of $k1 = 1.2$ and $b = 0.75$, in terms of baseline performance, Okapi BM25 does better than language modeling on description queries. However, when given estimated

[1]Note, the terminology here differs from Zhao and Callan's (2010). They used Multinomial RM to refer to the absolute tf weighting method.

true recall probabilities, BM25 does not perform as well as language modeling. Further experiments showed that the reason for the poorer performance of BM25 when true recall probabilities are given is that the default BM25 parameter values are not optimal. Setting $k1$ to 0.9 and $b$ to 0.5 gives the "Okapi tuned" column of Table 6.4. The performance of BM25 becomes comparable to that of the Dirichlet language model, though still slightly lower for 5 out of 7 test datasets. Since the results are not statistically significantly different, they do *not* mean that Language Modeling is better.

### 6.2.4 Predicted $\hat{P}(t|R)$ Weighting

Table 6.5: Retrieval performance (MAP) using predicted recall on long queries. Bold face means significant by both significance tests, two tailed, paired, with $p< 0.05$.

| TREC train | Test/ x-validation | LM *desc* | | Improvement | $p$ - randomization | $p$ - sign test |
|---|---|---|---|---|---|---|
| | | Baseline | $P(t\|R)$ | | | |
| 3sets | 4 | 0.1789 | **0.2261** | **26.38%** | 0.0000 | 0.0000 |
| 3-5 | 6 | 0.1586 | **0.1959** | **23.52%** | 0.0023 | 0.0002 |
| 3-7 | 8 | 0.1923 | **0.2314** | **20.33%** | 0.0043 | 0.0013 |
| 7 | 8 | 0.1923 | **0.2333** | **21.32%** | 0.0003 | 0.0077 |
| 3-9 | 10 | 0.1627 | 0.1813 | 11.43% | 0.0050 | 0.0595 |
| 9 | 10 | 0.1627 | **0.1810** | **11.25%** | 0.0012 | 0.0005 |
| 11 | 12 | 0.0239 | **0.0597** | **149.8%** | 0.0012 | 0.0000 |
| 13 | 14 | 0.1789 | **0.2233** | **24.82%** | 0.0001 | 0.0325 |

This section tests whether term recall can be effectively predicted to improve retrieval. In Table 6.5, we present results using predicted recall values (Chapter 5) as user term weights. Prediction features used here are idf, term centrality, concept centrality, replaceability and abstractness. Models were trained on TREC queries from previous year(s), and were tested using 5-fold cross validation on the 50 TREC queries of the next year. This means, the RBF support vector regression model was always trained on the 50 training queries (if training set includes only one TREC dataset). 50 test queries were split into 5 folds, 4 of which were used as development set to tune meta-parameters, 1 fold was used for testing. However, the learning model does not require that much development data. 2-fold cross validation uses fewer (only 25) development queries, and still yields the same performance and optimal parameter values as 5-fold cross validation does.

These results show that even though this initial set of features designed in this thesis can be noisy, they can still reliably predict term recall, resulting in a fair amount of significant gains in the retrieval for long queries. More detailed results and discussions are shown below, comparing predicted term recall term weighting with several state of the art retrieval models and retrieval techniques.

#### 6.2.4.1 Parameter Sensitivity and Tuning

The number of feedback documents - $n$ plays an important role in adapting the SVD features to different collections. During cross validation, when tuning $n$ on the development set, the optimal $n$ is found to be 180 on (smaller) TREC 3-8 datasets. For the larger WT10g of TREC 9-10, $n$ is 200. For the even larger .GOV TREC 11-12, $n$ increases to 500. For .GOV2 of TREC 13-14, $n$ is 600. The larger collections probably contain a larger number of helpful or even relevant documents, thus as expected, the optimal number of feedback documents increases as collection size does. $n$ might also depend on how many relevant documents there are for the query, and whether the majority sense of a query term is used. The

other two parameters – $m$, the number of latent dimensions to keep after SVD, and $\gamma$, which controls the kernel width – also affect performance, but the optimal values do not vary much across collections. On all datasets, cross validation found the two parameters to be within 130 to 170, and 1.2 to 1.6. Overall, the prediction accuracy is robust with regard to small perturbations of the three parameters. In the evaluations below, they are fixed at 150 and 1.5.

#### 6.2.4.2 Weighting v.s. No Weighting

The first thing to observe in Table 6.5 is that consistent improvements of 10% to 25% are observed in MAP across all 6 ad-hoc retrieval datasets, statistically significant by both sign and randomization tests. Most runs have strong significance levels of $p < 0.005$. This shows that models trained on one dataset can reliably predict term recall on a similar dataset. It also shows that the prediction model and the features do adapt to collections of different scales, with different levels of judgment depths, and do predict recall reliably.

Table 6.6: Retrieval performance (top precision) using predicted recall on long queries. Bold face means significant by both significance tests with $p< 0.05$.

| TREC train sets | Test/x-validation | LM *desc* | | | | | |
|---|---|---|---|---|---|---|---|
| | | Prec@10 | | | Prec@20 | | |
| | | Baseline | $P(t\|R)$ | Improve | Baseline | $P(t\|R)$ | Improve |
| 3 | 4 | 0.4160 | **0.4940** | 18.8% | 0.3450 | **0.4180** | 21.2% |
| 3-5 | 6 | 0.2980 | 0.3420 | 14.8% | 0.2440 | **0.2900** | 18.9% |
| 3-7 | 8 | 0.3860 | 0.4220 | 9.33% | 0.3310 | 0.3540 | 6.95% |
| 7 | 8 | 0.3860 | 0.4380 | 13.5% | 0.3310 | 0.3610 | 9.06% |
| 3-9 | 10 | 0.3180 | 0.3280 | 3.14% | 0.2400 | **0.2790** | 16.3% |
| 9 | 10 | 0.3180 | 0.3400 | 6.92% | 0.2400 | **0.2810** | 17.1% |
| 11 | 12 | 0.0200 | **0.0467** | 134% | 0.0211 | **0.0411** | 94.8% |
| 13 | 14 | 0.4720 | 0.5360 | 13.6% | 0.4460 | 0.5030 | 12.8% |

As predicted by the pathology of the emphasis problem (Section 3.6.1), with better term weighting, top precision should also improve, even though the model parameters are tuned on MAP instead of top precision. Table 6.6 shows a 10% to 25% gain in top precision across all collections. However, most of these improvements are not statistically significant as measured by the sign test. The randomization test takes into account the relative gains instead of just the sign of the difference between the treatment and the baseline runs. Measured by the randomization test, more collections show significant gain in top precision. Another observation is that more collections show significant gain when measured by P@20 than when measured by P@10. These insignificant improvements at top precision on individual collections indicate that improvements for the top few documents is still not very stable. However, given the large improvements on average, and given that the large improvements are consistent across all 6 collections, we believe that a larger set of test queries would provide enough statistical power to observe the significance of the gain in top precision.

#### 6.2.4.3 Weighting Short v.s. Weighting Long Queries

Table 6.7 shows that on short queries, reweighting original query terms by their predicted recall values does perform slightly better than the unweighted title baseline, but it does not give a significant improvement on most datasets as over the long (description) query baseline. This result is also consistent with the

Table 6.7: Retrieval performance for predicted-recall based term weighting, trained and tested on short (*title*) queries. Bold face means significance over LM *title* baseline by both tests ($p < 0.05$).

| TREC train sets | Test/x-validation | LM *title* - Baseline | LM *title* - $P(t\|R)$ | $p$ - randomization | $p$ - sign test |
|---|---|---|---|---|---|
| 3 | 4 | 0.1789 | **0.2199** | 0.0002 | 0.0002 |
| 3-5 | 6 | 0.2362 | 0.2386 | 0.2102 | 0.0106 |
| 3-7 | 8 | 0.2518 | 0.2261 | 0.9516 | 0.9605 |
| 7 | 8 | 0.2518 | 0.2512 | 0.5557 | 0.1510 |
| 3-9 | 10 | 0.1577 | 0.1643 | 0.1654 | 0.3179 |
| 9 | 10 | 0.1577 | 0.1706 | 0.0260 | 0.3179 |
| 11 | 12 | 0.0964 | 0.0957 | 0.7818 | 0.3089 |
| 13 | 14 | 0.2511 | 0.2525 | 0.2769 | 0.1562 |

upper-bound term weighting runs shown in Table 6.2, which shows that true term recall term weighting on short queries does not see as much an improvement as applied on long queries.

There are several reasons for this difference between reweighting short v.s. long queries. Firstly, reweighting the only two terms from a short query is the same as classifying documents according to only 2 features (terms): no matter how excellent term weighting is, the classifier won't achieve the same discrimination power as with more features (query terms)[2]. Secondly, the baseline retrieval model does not handle long queries as well as the short queries. The reason is that being idf-centric, for long queries, the baseline retrieval model shares a larger risk of focusing on several of the high idf but low recall terms in the query, ignoring other high recall but low idf terms, causing the emphasis problem and hurting both top precision and recall. The error analyses performed in the RIA workshop (Harman and Buckley 2009) also confirms that this is one of the main causes of failure.

#### 6.2.4.4 Language Model v.s. Okapi BM25

Table 6.8: Retrieval performance (MAP) of language model and Okapi BM25 with *predicted* recall weighted query terms. BM25 parameters $k1 = 0.9$, $b = 0.5$. Bold face means significant by both randomization and sign tests with significance level $p< 0.05$. * means significant only by randomization test with significance level $p< 0.05$.

| TREC train sets | Test/x-validation | LM *desc* MAP | | | Okapi *desc* MAP | | |
|---|---|---|---|---|---|---|---|
| | | Baseline | $P(t\|R)$ | Improve | Baseline | $P(t\|R)$ | Improve |
| 3 | 4 | 0.1789 | **0.2261** | **26.4%** | 0.2022 | 0.2113* | 4.50%* |
| 3-5 | 6 | 0.1586 | **0.1959** | **23.5%** | 0.1776 | **0.2052** | **15.5%** |
| 3-7 | 8 | 0.1923 | **0.2314** | **20.3%** | 0.2239 | 0.2298 | 2.64% |
| 7 | 8 | 0.1923 | **0.2333** | **21.3%** | 0.2239 | 0.2314 | 3.35% |
| 3-9 | 10 | 0.1627 | 0.1813 | 11.4% | 0.1680 | 0.1770 | 5.36% |
| 9 | 10 | 0.1627 | **0.1810** | **11.3%** | 0.1680 | 0.1709 | 1.73% |
| 11 | 12 | 0.0239 | **0.0597** | **150%** | 0.0406 | 0.0333 | -18.0% |
| 13 | 14 | 0.1789 | **0.2233** | **24.8%** | 0.2171 | 0.2247* | 3.50%* |

[2]A weighted query can be seen as an efficient classifier to classify the whole collection. See for example (Anagnostopoulos, Broder, and Punera 2006; Zhu, Zhao, Callan, and Carbonell 2008). In both of these cases, more than 10 terms/features are needed to obtain optimal performance.

Section 6.2.3.3 shows that with true term recall probabilities, Language Modeling performs slightly better than Okapi BM25. Because of the higher upper-bound performance of Language Modeling, we observe similarly that with predicted-recall based term weighting that Language Modeling performs slightly better than Okapi BM25, as shown in Table 6.8.

#### 6.2.4.5 New Features v.s. Traditional Features

Table 6.9: Retrieval performance (MAP) using predicted recall - traditional v.s. new features. Bold face means significant by both significance tests with $p< 0.05$.

| TREC train sets | Test/x-validation | LM *desc* | | | |
|---|---|---|---|---|---|
| | | Baseline | $P(t\|R)$ | Idf | Clarity |
| 3 | 4 | 0.1789 | **0.2261** | 0.1776 | 0.1900 |
| 3-5 | 6 | 0.1586 | **0.1959** | 0.1691 | **0.2028** |
| 3-7 | 8 | 0.1923 | **0.2314** | **0.2325** | **0.2309** |
| 7 | 8 | 0.1923 | **0.2333** | **0.2383** | **0.2318** |
| 3-9 | 10 | 0.1627 | 0.1813 | 0.1696 | 0.1399 |
| 9 | 10 | 0.1627 | **0.1810** | 0.1687 | 0.1516 |
| 11 | 12 | 0.0239 | **0.0597** | **0.0576** | **0.0473** |
| 13 | 14 | 0.1789 | **0.2233** | 0.2227 | 0.2057 |

Traditionally, only idf has been used as the feature to predict $P(t|R)$ (Greiff 1998; Metzler 2008). To provide a baseline for comparing with the new features, we trained an RBF support vector regression model on the same training sets using idf as the only feature (the "Idf" row of Table 6.9).

Since we are using a powerful non-linear regression method to predict term recall, and overfiting is unlikely given the hundreds of training samples with only one feature, we do not expect any of the prior models of linear or 3-piece linear functions of idf (Greiff 1998; Metzler 2008) to be better than the RBF support vector regression baselines.

Compared to the new features obtaining stable and significant improvement over the Language Model baseline, predictions based only on idf results in an unstable gain. Across most collections, performance improvement is not significant, and in one collection performance is worse than that of the baseline language model. Only 3 out of the 8 cases show statistically significant improvement.

Clarity is traditionally used to predict query performance, thus we include another baseline that predicts recall based on term clarity as the only feature (row "Clarity" of Table 6.9). Performance is also unstable across collections, and on average is similar to using only idf to predict.

#### 6.2.4.6 Comparing to the Relevance Model

The Relevance Model theory (Lavrenko and Croft 2001) suggests to use term relevance probabilities as user term weights, which we follow, as shown in Section 6.1.2. The Relevance Model also provides a way to estimate the relevance based term weights from top documents of an initial retrieval. This estimation procedure, though different in goal, is similar to how we created the SVD features using top ranked documents of an initial retrieval. Given the similarities, we also include the Relevance Model as another baseline.

The Relevance Model used here is the Query Likelihood Relevance Model (QLRM) by Metzler et al. (2005), or equivalently the RM3 model (Lv and Zhai 2009), which introduces expansion terms into the query, reweights the expansion terms with the estimated probability of relevance ($P_m(t|R)$), and linearly

Table 6.10: Retrieval performance (MAP) using predicted recall - supervised v.s. unsupervised term weighting. The RM Reweight-Only run uses unsupervised Relevance Model weights, while the RM Reweight-Trained run uses supervised term weights. The RM new+RMw-Trained run uses the new set of 5 features together with the Relevance Model weight, a total of 6 features for supervised $P(t|R)$ prediction. Bold face means significantly better than the Language Model (LM *desc*) baseline by both significance tests with $p< 0.05$.

| TREC train/test sets | LM *desc* | | RM *desc* | RM *title* | RM *desc* | RM *desc* | RM *desc* |
|---|---|---|---|---|---|---|---|
| | Baseline | $P(t\|R)$ | Baseline | Baseline | Reweight-Only | Reweight-Trained | new+RMw-Trained |
| 3/4 | 0.1789 | **0.2261** | **0.2423** | N/A | **0.2215** | **0.2322** | **0.2369** |
| 3-5/6 | 0.1586 | **0.1959** | 0.1799 | 0.2605 | 0.1705 | **0.1954** | **0.2010** |
| 3-7/8 | 0.1923 | **0.2314** | **0.2352** | **0.2790** | **0.2435** | **0.2551** | **0.2456** |
| 7/8 | 0.1923 | **0.2333** | **0.2352** | **0.2790** | **0.2435** | **0.2556** | **0.2509** |
| 3-9/10 | 0.1627 | 0.1813 | **0.1888** | 0.1710 | 0.1700 | 0.1901 | **0.1854** |
| 9/10 | 0.1627 | **0.1810** | **0.1888** | 0.1710 | 0.1700 | **0.1861** | **0.1870** |
| 11/12 | 0.0239 | **0.0597** | 0.0221 | 0.0896 | **0.0692** | **0.0555** | **0.0616** |
| 13/14 | 0.1789 | **0.2233** | 0.1774 | **0.2893** | 0.1945 | **0.2282** | **0.2279** |

interpolates the expansion query with the original query to produce a final ranking. Term recall based term weighting, on the other hand, only reweights the original query terms: no expansion terms are introduced, and no interpolation is done with the original query. Thus, to make the two methods comparable, we create a baseline that weakens the Relevance Model to only use the term weights ($P_m(t|R)$) to reweight the original query terms, called *RM Reweight-Only* in Table 6.10. This baseline uses the term weights estimated by the Relevance Model from top-ranked documents to only re-weight the original query terms, and is more similar to our approach of using top-ranked documents to compute user term weights for the original query terms.

Results in Table 6.10 show RM Reweight-Only to be unstable. On some collections, it is significantly better than the Language Model baseline, but on as many other collections, the improvement is insignificant by both tests. This is expected, because relevance models can be seen as directly using term centrality to determine term weights, thus, it shall not outperform the prediction based on the whole set of features. In some cases, RM Reweight-Only outperforms recall prediction. This shows that favoring higher ranked documents according to their relevance scores, as the Relevance Model does, has some advantages, whereas our current local SVD treats all top-ranked documents the same.

RM Reweight-Only performs just slightly lower than the RM baseline which includes expansion terms. This means two things, first, *expansion terms in the Relevance Model contribute little to retrieval for queries of 5 or more words long*, and second, *expanding terms as a weighted combination of feedback terms, as in the Relevance Model or Rocchio feedback, may not be particularly effective for when applying pseudo relevance feedback on long queries.*

RM new+RMw-Trained is top performing. This shows that the new set of 5 features for $P(t|R)$ prediction and the Relevance Model weights both contain useful information for $P(t|R)$ prediction that can be combined to achieve higher retrieval performance than using them separately.

#### 6.2.4.7 Supervised Weighting v.s. Unsupervised RM *Expansion*

The Relevance Model RM3 interpolates the weighted expansion query with the original query. The same development sets were used to tune the free parameters (number of feedback documents, feedback terms,

and mixing weight with the original query). The same cross validated results were reported.

One interesting observation is that, as shown in Table 6.10, despite a more expressive feedback query (weighted expansion terms + interpolation with original query), trained on the same datasets, the Relevance Model baseline is unstable compared to predicted recall weighting. It is significantly better than the baseline on 3 collections, but insignificant on 3 others. It hurts performance for the low initial retrieval run of TREC 12. This performance is expected because even though the Relevance Model estimates relevance based term weights, it estimates term weights in an unsupervised fashion, while our supervised framework uses relevance judgments from training queries to guide the prediction. Another difference is that *the Relevance Model tends to use only a few top documents (around 5-20), while our local SVD performs optimally with hundreds of documents. The SVD based features can leverage term appearance information from more documents, achieving a more stable performance.*

Overall, the results from this section and the previous section suggest that *for description queries, most of the performance gain from pseudo relevance feedback (PRF) methods is due to term weighting instead of expansion terms, and due to the fact that term weights computed from PRF methods tend to correlate well with true recall*. The reason is that *terms that occur consistently in relevant documents also tend to appear consistently in top ranked documents*. In fact, a 0.63 correlation is observed on TREC 4 test set, while our best predicted recall have a higher 0.80 correlation with recall (Table 5.2), explaining the better performance.

On title queries, however, because of the more accurate user query, better initial retrieval performance and the introduction of weighted expansion terms, the Relevance Model (Table 6.10) outperforms true recall weighted title queries on 4 out of the 6 collections, and in most cases outperforms predicted recall weighted description queries (bold faced entries in the "RM *title*" column are the runs that perform significantly better than the title query baseline). This observation is consistent with the commonly known property of pseudo relevance feedback methods that the performance of pseudo feedback methods (e.g. the Relevance Model) depends largely on the quality of the initial query.

#### 6.2.4.8 Supervised Weighting v.s. Unsupervised RM *Weighting* – using Relevance Model Weight as a Feature

To understand why the supervised recall predictions perform better than the Relevance Model term weights, and to further show that it is truly supervised recall prediction that is working here, instead of just a clever use of the pseudo relevance feedback information from the top ranked documents, we designed an experiment using RM term weights as the only feature to predict recall. The results are presented in Table 6.10 as the RM Reweight-Trained runs. The RM Reweight-Trained runs used Relevance Model term weights as the only feature and trained a recall prediction model on the training set to predict term recall on the test set. When generating training data, a simple per query scaling is used to map the near-zero RM weights to [0, 1], so that the maximum query term weight is always 1 for a given query. Without scaling, the RM term weights are extremely small, and vary wildly across queries. For a discussion of different normalization schemes see Section 5.2.2.7.

RM Reweight-Trained performed consistently better than the Reweight-Only method, except on TREC 12. *This consistent improvement directly shows that it is the supervised learning of term recall that is improving the unsupervised Relevance Model estimates, not just another better use of the pseudo relevance feedback information.* The Reweight-Trained method performs slightly better than the predicted recall term weight on average, though not on every dataset. This suggests that the RM weight can be an effective feature for predicting recall.

The RM Reweight-Trained model is a function that maps the RM term weights into better performing recall values. Thus, we can observe where the Relevance Model is performing suboptimally by directly

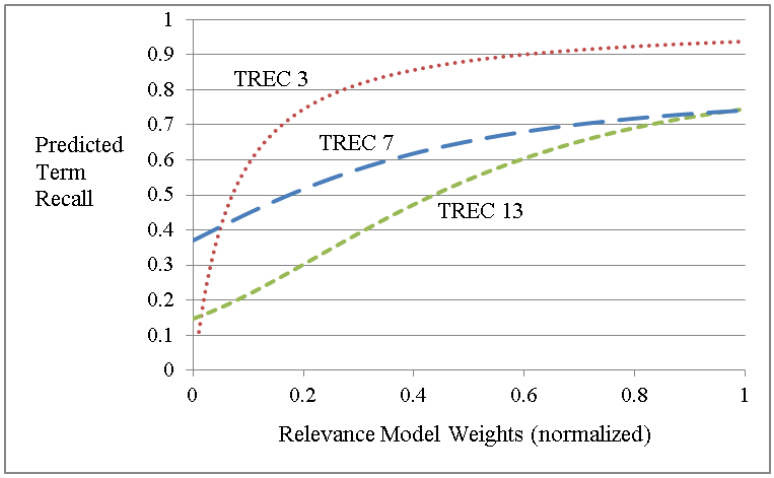



Figure 6.1: The learned functions using unsupervised Relevance Model term weights (x-axis) alone to predict supervised term recall (y-axis). The 3 lines correspond to the models learnt on TREC 3, 7 and 13 datasets with description/long queries.

looking at the function. Three such functions trained on the TREC 3, 7 and 13 datasets are plotted in Figure 6.1. They all downplay larger weights, and boost terms with lower middle range weights (0-0.6) to larger values. The reason why this mapping works is perhaps because of the following two cases where the Relevance Model generates biased term weight estimates. *Firstly, the Relevance Model tends to be skewed by the extremely frequent (bursty) occurrences of some popular terms, and consequently overestimates the recall values of these frequent terms. Secondly, the pseudo relevance feedback method typically depends on about 10 top-ranking documents to estimate those weights. This top document set can be easily biased to some query terms, but not others, causing the underestimation for these other query terms.*

In terms of feature effectiveness, the relevance model weights perform quite well. Using this feature alone gets a performance that is similar to using the 5 new features. The RM Reweight-Trained method performed similarly as the Recall run using the 5 new features, except on the TREC 6 test collection, where the statistical significance dropped below the threshold. Additionally, the Relevance Model typically used at most tens of feedback documents, and the meta-parameters (either the optimal number of feedback documents or the gamma parameter for the RBF kernel in the regression model) varied a lot during cross-validation on two out of the 8 test collections, while the local LSI based method performs best with hundreds of feedback documents and the model parameters are more stable. This suggests that *the Relevance Model relies heavily on the pseudo-relevant documents to be actually relevant, while for our recall prediction features, local LSI is only used to find synonyms of the query terms in a query dependent way, thus does not rely much on whether the feedback documents are actually relevant, as long as they are roughly about the same word senses that are being used in the query*.

The run RM new+RMw-Trained includes Relevance Model weight (RMw) as the 6th feature, and further improves MAP by about 5% over the 5 feature based prediction and by about 2% over the Relevance Model weight prediction. However, the meta-parameters become less stable during cross validation, a sign of overfitting. To avoid overfitting, in this experiment with all 6 features, the meta-parameters for the 5 features are fixed according to earlier experiments that only used these 5 features, and the only tuning parameter during cross validation is the number of top ranked documents to compute the Relevance Model weights. Overall, the RM new+RMw-Trained runs show that the Relevance Model weight is an effective feature for $P(t|R)$ prediction, and provides useful information even in addition to the 5 features that we

proposed.

#### 6.2.4.9 Full Feature Set v.s. Feature Subsets for Prediction

Table 6.11: Effects of features on TREC 4. Bold face means significance over LM baseline by both tests ($p < 0.005$).

| Features used | MAP | Features used | MAP |
|---|---|---|---|
| Idf only | 0.1776 | All 5 features | **0.2261** |
| Idf + Term Centrality | **0.2076** | All but Term Centrality | **0.2235** |
| Idf + Concept Centrality | **0.2129** | All but Concept Centrality | **0.2066** |
| Idf + Replaceability | 0.1699 | All but Replaceability | **0.2211** |
| Idf + Abstractness | 0.1900 | All but Abstractness | **0.2226** |

To see the effects of the individual features and how they interact, in Table 6.11 we present a feature ablation study, showing retrieval performance for different feature combinations using TREC 3 as training and TREC 4 as test set. Results show that all the features contribute to effective retrieval. Among them, the Concept Centrality feature is most effective and results in more stable improvements, but the other features can still significantly improve retrieval without Concept Centrality. Similar trends are observed on the other test sets. The only exception is the TREC 10 test set, where Term Centrality outperforms Concept Centrality, and the retrieval performance gains from prediction based on the Concept Centrality feature alone is not significant by the sign test.

*Overall, the evaluation using L1-loss of recall prediction (Table 5.3) and feature correlation (Table 5.2) show very similar trends as retrieval effectiveness (Table 6.11).* IDF alone is not effective in predicting recall. All the 5 designed features contribute to recall prediction accuracy, and further, retrieval effectiveness. *For the current set of features, better recall prediction leads to better retrieval performance. The true/oracle recall values lead to the best retrieval performance (the true recall weighting runs in Table 6.1).*

## 6.3 Summary

This chapter shows that on 6 ad hoc retrieval test collections, term weighting based on the *true* recall probabilities improve retrieval performance (MAP) of the modern retrieval models (Okapi BM25, and language models) by 30% to 80%. This dramatic gain shows the potential of using term recall based term weighting, and means that a prediction method does not need to be optimal to achieve significant gains. If including term expansion methods guided by term mismatch probabilities, the potential gain would be even greater.

A 15% to 25% gain in MAP is observed across the 6 test collections when using the 2-pass predicted recall probabilities as user term weights. A similar gain in top precision is also observed. This improvement is larger than what a strong pseudo relevance feedback baseline can offer, as shown by experiments with the Relevance Model.

This set of experiments shows direct evidence that term mismatch is an important problem to solve in order to improve ad hoc retrieval performance. Term weighting is one of the simplest retrieval interventions allowed by the term mismatch predictions. The effectiveness of term weighting shows that more advanced interventions to solve term mismatch could be even more promising for improving retrieval

performance. One such advanced intervention is mismatch guided query term expansion discussed in the chapter below.

Since the 2-pass prediction approach outlined in this chapter is quite expensive to use, the chapter below aims to apply the data analyses on the variation of the $P(t|R)$ probability to understand whether these expensive 2-pass features are really necessary for effective prediction, and to design more efficient $P(t|R)$ prediction methods.

# Chapter 7

# Predicting $P(t|R)$ Variation & Efficient $P(t|R)$ 1-pass Prediction

Data analysis in Chapter 4 showed that query dependent features such as the association level of the term with the query are useful for effective $P(t|R)$ prediction. At the same time, many repeating term occurrences share similar $P(t|R)$ values across different queries. These can be used to develop a more efficient method for predicting $P(t|R)$ using historical data. Retrieval experiments show that utilizing the information about prior occurrences of the same term (e.g. from the training set or a query log without any judgments or click-through) to predict the probability $P(t|R)$ in a new query leads to a performance close to that of the 2-pass model presented in Chapter 5, but with far lower computational complexity. This makes it possible to use $P(t|R)$ predictions in real time response scenarios.

## 7.1 Background

The query dependent features used in Chapter 5 are quite expensive to compute: 3 out of the 5 features require an initial retrieval and Latent Semantic Indexing over the top ranked documents ('local LSI'). We refer to this approach as the 2-pass method, because of its two retrieval steps. The inefficiency of the 2-pass approach makes it impractical for low-latency search environments.

At the same time, data analyses on TREC 3 to 7 datasets (Chapter 4) have demonstrated that the $P(t|R)$ probability have a small variation for 62% of the term occurrences for the same term occurring in different queries. This means the query dependent features may not need to be very complex to capture the variation. This chapter designs computationally efficient query dependent features based on the causes of the query dependent $P(t|R)$ variation identified in Chapter 4, and proposes novel prediction methods that use information from prior occurrences of a term to predict the $P(t|R)$ probability of the same term in a new query.

The goal of this chapter is to produce a $P(t|R)$ prediction method that is 1-pass only for a new query, and can still achieve much of effectiveness of the 2-pass method.

This overall goal is broken down into two stages. The first stage efficiently predicts the $P(t|R)$ variation for the same term in different queries, and is discussed in Section 7.2. The second stage utilizes an instance based learning framework to render a single $P(t|R)$ prediction based on multiple historic occurrences of a test term and their predicted $P(t|R)$ differences with the test occurrence. The overall architecture of this instance based learning framework is explained in Section 7.3.

## 7.2 Modeling and Predicting $P(t|R)$ Variation

This section explains the prediction framework used to predict the term recall difference $P(t|R_{q1}) - P(t|R_{q2})$ between two occurrences of the same term $t$ in two different queries $q_1$ and $q_2$. This includes the design of the features, the training data generation process and the learning model. The efficient prediction of $P(t|R)$ variations provides the possibility to efficiently predict $P(t|R)$ based on historic occurrences of the term, which is discussed in detail in Section 7.3.

Overall, this prediction approach relies on the same training datasets with queries and relevance judgments as the 2-pass prediction method presented in Chapter 5; no extra source of information is needed.

### 7.2.1 Problem Formulation

We formulate $P(t|R)$ variation prediction as a regression problem, where the goal is to predict the $P(t|R)$ difference between the current and a previous occurrence of the term.

Given two occurrences $t_{q1}$ and $t_{q2}$ of the same term $t$ in two training queries $q_1$ and $q_2$, assume $P(t|R_{q1}) = N_1 \in [0,1]$, and $P(t|R_{q2}) = N_2 \in [0,1]$, then we aim to use features generated from $< t_{q1}, t_{q2}, q_1, q_2, C >$ to predict the difference in $P(t|R)$, i.e. $\Delta P(t|R_x)|_{q2}^{q1} = (N_1 - N_2)$. Here, $C$ is the document collection.

Alternatively, one can treat this as a classification task, where the target to learn is whether a current occurrence of the term will have a very different $P(t|R)$ probability from that of a prior occurrence. When the classifier decides the difference to be small, during retrieval, the $P(t|R)$ from the prior occurrence of the term is directly used to weight the term in the new query. However, this classification setup has several disadvantages. The fundamental flaw is that we are interested in the $P(t|R)$ probabilities, thus, we want to know the exact predicted difference, not just a decision of yes/no. Another disadvantage is that when the classifier decides otherwise, i.e. a large difference from a previous occurrence, the prior occurrence cannot be used to predict $P(t|R)$ of the current occurrence, and the prediction may need to fall back to the expensive 2-pass approach. Furthermore, the definition of a *small* difference is unclear. Would 0.1 be a small enough difference? Should 0.2 be considered a small difference? Using any threshold here would be ad hoc.

### 7.2.2 Features – Term Occurrence Pair

The features are designed based on pairs of term occurrences. These features are designed to reflect the causes of the query dependent $P(t|R)$ variation examined in the data analyses in Section 4.2. The goal is to have them correlate with the prediction target – the *variation* in the $P(t|R)$ probabilities of $t$ in two different queries.

#### 7.2.2.1 Part Of Speech

Since a change in the Part Of Speech (POS) tag of a word sometimes indicates a change in word sense, we use the POS tag of a query term (using the Brill tagger) to generate the first set of features. These are 22 binary features of whether a query term is of a certain POS tag, e.g. noun, adjective, verb etc. (We do not make further distinctions beyond these main POS categories. Trying to reduce the 22 features down to fewer more effective ones did not help performance.) Suppose $f_{NN}(t_{q1})$ is the feature that indicates whether term $t_{q1}$ is a noun in the query it appears in, then the final feature for the two occurrences $t_{q1}$ and $t_{q2}$ is defined as difference $f_{NN}(t_{q1}, t_{q2}) = f_{NN}(t_{q1}) - f_{NN}(t_{q2})$. Lease et al. (2009) similarly used POS features in predicting term weights.

#### 7.2.2.2 Clarity

Word clarity (Cronen-Townsend et al. 2002) can be used as a measure of word abstractness, where each word is treated as an individual query and query clarity is measured according to (Cronen-Townsend et al. 2002). Abstract words, e.g. "disorder" in Section 4.2, may have a high variation in $P(t|R)$ in different queries. This is because an abstract term by itself may have a low $P(t|R)$, but may have a high $P(t|R)$ probability when it appears in a fixed phrase.

#### 7.2.2.3 Idf

The inverse document frequency (idf) of a term indicates whether a term is likely content-bearing. Idf is also traditionally used to predict $P(t|R)$, and weakly correlates with $P(t|R)$. We use the same form of idf as in Chapter 5: $log((N - df)/df)$ where $N$ is the total number of documents in the collection and $df$ is the document frequency of the term. This is also the idf part of the RSJ weight (Robertson and Spärck Jones 1976).

#### 7.2.2.4 Previous $P(t|R)$

Assuming, we are predicting the $P(t|R_{q2})$ of $t$ occurring in query $q_2$, using the occurrence of $t$ in the training query $q_1$, the known probability $P(t|R_{q1})$ calculated from the relevance judgments of the training $q_1$ can also be used as a feature. It is called previousP. This feature allows the prediction algorithm to learn that the prediction target – the difference between two probabilities – is constrained, for example, when $P(t|R_{q1})$ is already close to 1, the $P(t|R_{q2})$ (being also a probability) won't be much larger than that of $P(t|R_{q1})$.

#### 7.2.2.5 Phrase Features

According to the data analysis of Section 4.2, term use differences contribute to $P(t|R)$ variation. One particular example of a difference in term use is whether a term appears in a phrase or not. Thus a statistical phrase feature is designed to measure how likely a term is part of a phrase in the query. To construct this feature for the $n$th term in the query, we compute how likely terms $n-1$ and $n$, or terms $n$ and $n+1$ are to be part of a phrase. The first and the last term of the query only have one of the above two cases. To measure whether two terms form part of a phrase, we use the probability of observing two terms consecutively in a document given that the two terms appeared in the document. More formally, define the probability of $t_1$-$t_2$ being *part* of a phrase to be

$$Phrase(t_1, t_2) = \frac{|\{d|bigram(t_1, t_2) \in d\}| + 1}{|\{d|t_1 \in d \& t_2 \in d\}| + 2} \tag{7.1}$$

Then, the probability of $t_n$ being part of a phrase is

$$phrase(t_n) = 1 - (1 - Phrase(t_{n-1}, t_n)) \cdot (1 - Phrase(t_n, t_{n+1})) \tag{7.2}$$

The final form of this phrase feature is the difference of the phrase probabilities of the two occurrences of the term. (The document counts for the bigrams and term conjunctions need additional retrieval steps, however, these can be easily parallelized and no document vector needs to be pulled out as in pseudo-relevance feedback.)

Directly computing this feature on the fly requires another retrieval step – intersecting inverted lists. Offline computation of the collection frequencies of common bigrams and caching them can move most of

this online computation to offline. This feature was tested in our experiments, but not found to be among the most effective features.

#### 7.2.2.6 Query Association Features

The phrase feature can be thought of as a specific measure of association between the query term and its local context in the query. But more generally, in order to capture the association variations of the $P(t|R)$ probability as we've seen in Sections 4.2 and 4.3, new term-query association measures need to be designed. The local LSI based features from Section 5.2.2 are such measures. Here, we need something more efficient. The easiest of such association measures would be based on the term similarities between one query term and all the other query terms.

For simplicity, we compute term-term similarity as the cosine similarity of the two document vectors corresponding to the two terms. The document vector for a term consists of term frequency values for the term in all indexed documents. In this chapter, term frequency is computed as $log(1+tf)$, where $tf$ is the raw term frequency count. Because we are using cosine similarity, including the idf weights of the terms only changes the term vectors by a constant factor, and won't affect the final similarity score.

Given this definition of term similarity, the association between a query term and its query can be measured as the maximum or average or minimum of the pair-wise similarities between the term and all the other query terms.

$$Min\_assoc(t_i, q) = min_{j \neq i, t_j \in q} cosine(t_i, t_j) \tag{7.3}$$

$$Avg\_assoc(t_i, q) = avg_{j \neq i, t_j \in q} cosine(t_i, t_j) \tag{7.4}$$

$$Max\_assoc(t_i, q) = max_{j \neq i, t_j \in q} cosine(t_i, t_j) \tag{7.5}$$

For a pair of repeating occurrences of a term to predict $P(t|R)$ variation, the final feature value is the difference between the two occurrences. The pair-wise term-term cosine similarities (at least for frequent term pairs) can be computed offline to speed up online performance. The term-term similarities can be computed on a sampled/smaller collection to further save computation.

#### 7.2.2.7 Derived Comparison-based Features

Two additional features derived from the above basic features are also tested. Binary features of whether the two words are exactly the same in the two queries before stemming, and whether the two words have the same POS. These two features are denoted as eqWord and eqPOS.

#### 7.2.2.8 Dependency Parse Feature

As in Chapter 5, the binary feature of whether a query term is a leaf node of the dependency parse (depleaf) of the original query is also tried. The difference is used for a term pair.

#### 7.2.2.9 Summary

Overall, POS (22 features), depleaf, eqWord, eqPOS, phrase, term-query association (3 possible features) and previousP are query dependent features, of which only previousP is derived solely from the previous occurrence, and all the other features are based on the pair of term occurrences. All the rest of the features such as idf and clarity are term specific and query independent. The term specific features do not depend on the query, thus can be pre-computed and stored for faster online performance.

### 7.2.3 Training Instance Generation

Each training instance is generated from an ordered pair of term occurrences, e.g. using $t$ in $q_1$ and $q_2$ to predict the $P(t|R)$ difference between them $\Delta P(t|R_x)|_{q_2}^{q_1} = (N_1 - N_2)$. Because of that, a number of different ways can be used to generate training instances. Take for example $n$ occurrences of the term $t$ (in $n$ different queries).

The first method goes through the whole set of training queries once, sequentially, caching exactly one prior occurrence for each unique term, and when the next occurrence for the term is observed, the pair is used to generate one training instance. For $n$ occurrences of the same term, exactly $n - 1$ training instances are created. We call this the sequential method.

The second method is very similar to the sequential method, but instead goes through the collection twice, first to last and last to first, generating $2 * (n - 1)$ training instances using the consecutive pairs of term occurrences. We call this the back-forth method. This method generates twice as many training samples from the same training set, and increases prediction accuracy over the sequential method.

The third method is called the all-pairs method which uses all the $n * (n - 1)$ pairs of occurrences to generate training instances. Pilot experiments show that the all-pairs method biases toward frequently occurring terms, and does not improve prediction performance any further.

Experiments reported here only show results from the back-forth method.

### 7.2.4 Prediction Model

As in the case of $P(t|R)$ prediction in Section 5.2.4, an RBF support vector regression model is used for learning and prediction, because the features have non-linear relations with the prediction target.

## 7.3 Using Variation Prediction to Predict $P(t|R)$ - Overall Architecture

Given a training set and the $P(t|R)$ variation prediction framework setup above, how to predict $P(t|R)$ for a new occurrence of a query term that exists in the training set? This section describes the overall prediction and retrieval algorithm.

The training set used to train the 2-pass $P(t|R)$ prediction model (Chapter 5) is now used to train a $P(t|R)$ variation prediction model based on the repeating term pairs in the training set. For example, for the TREC 3 training set, 245 query terms, including repeating term occurrences, were used to train the 2-pass $P(t|R)$ prediction model, while in this chapter, only the 78 repeating term pairs are used to train the $P(t|R)$ variation prediction model. We use the back-forth method to generate training instances, so really there are only 39 term pairs if using the sequential method.

Given a test query, for all the new terms that do not appear as a query term in the training set (one of the 245), $\hat{P}(t|R)$ is simply assumed to be 0.5, which is roughly the training set average.

For a repeating term, given its test occurrence, and $n$ prior occurrences of the same term in the training set, $n$ test occurrence pairs can be constructed, with the test occurrence being $t_2$ and each training occurrence as $t_1$. (See Section 7.2.1 for the meaning of $t_1$ and $t_2$.) Each test pair will generate a separate $P(t|R)$ variation prediction, and in order to produce a single $\hat{P}(t|R)$ probability, a merging step is needed.

Several different merging strategies have been tried, for example, taking the mean of the predicted $\hat{P}(t|R)$ values, the arithmetic mean, the geometric mean or taking the prediction from the nearest neighbor (using the prediction that has the lowest predicted variation). The nearest neighbor method performs the best, and we only report the performance of this method. Suppose $m$ is the nearest neighbor, i.e.

$$m = argmin_i |\hat{V}(i, j)| \tag{7.6}$$

then,

$$\hat{P}(t|R_j) = P(t|R_m) - \hat{V}(m, j) \tag{7.7}$$

Here, $i$ is a historic occurrence of $t$ in $q_i$ with known $P(t|R_i)$, $\hat{V}(i, j)$ is the model predicted $P(t|R)$ variation given the historic and test occurrences: $i$ and $j$. The predicted $\hat{P}(t|R_j)$ is given by Equation 7.7, by assuming, $\hat{V}(i, j) = P(t|R_i) - \hat{P}(t|R_j)$, (following Section 7.2.1). When multiple historic occurrences have the same predicted difference with the test occurrence, i.e. same $\hat{V}(i, j)$, the average $\hat{P}(t|R_j) = avg_m[P(t|R_m) - \hat{V}(m, j)]$ is used. If a method always predicts $\hat{V}(i, j)$ to be 0, it is equivalent to taking the historic average: $\hat{P}(t|R_j) = avg_i P(t|R_i)$.

## 7.4 Experiments – Variation Prediction Accuracy

The Section 7.2 above presents the framework of extracting numeric features and generating training samples to predict the variation of the $P(t|R)$ probability – the difference between the two $P(t|R)$ probabilities of two different occurrences of the same word in two different queries. This section measures how accurately the regression model predicts the variation and how it impacts retrieval performance on a test set. Feature-target correlations are also presented to give a sense of how well each type of feature associates with the prediction target - $P(t|R)$ variation.

The gamma parameter that controls the RBF kernel width in the regression model is fixed at 1.5 from a pilot study. Other close-by choices do not make a large impact on prediction performance.

Table 7.1: $P(t|R)$ variation prediction error in average L1 loss (the smaller L1 loss, the better the prediction). Trained on TREC 3 repeating words and tested on TREC 4 repeating words. Retrieval performance using the $P(t|R)$ predictions as term weights is also shown for reference, measured in Mean Average Precision. Bold face shows top performing entries in each column.

| **Features for variation prediction** | **L1 loss** | **MAP** |
|---|---|---|
| No feature baseline (always predict 0-difference, term specific, query independent) | 0.1753 | 0.1889 |
| POS (alone) | **0.1769** | 0.1878 |
| previousP (alone) | 0.1769 | 0.1905 |
| POS+previousP | 0.2119 | 0.1925 |
| POS+previousP+Max_assoc+Avg_assoc | **0.1755** | **0.1930** |
| ... +idf+clarity | 0.1864 | **0.1930** |
| POS+previousP+idf+clarity | 0.2330 | **0.1928** |
| ... +eqWord+eqPOS+depleaf | 0.2178 | 0.1915 |
| ... ... +phrase | 0.2181 | 0.1913 |
| 2-pass $P(t|R)$ prediction | **0.1539** | **0.2261** |

Table 7.1 lists a set of different feature combinations and their corresponding prediction accuracy and retrieval performance. Because the performance differences we see here are not very large, we are looking for a run that gets both low prediction error and high retrieval performance, to avoid overfitting on either metric. Rows starting with a "+" sign add the feature(s) to the previous row. In terms of L1 loss for variation prediction, the baseline of always predicting 0 difference achieves a fairly low loss. This means that the $P(t|R)$ variations are usually close to 0. However, despite the low prediction error, the corresponding retrieval performance of 0.1889 is worse than the retrieval performance of 0.1930 using more features. *Using feature based variation prediction is better than directly weighting repeating terms using a previous $P(t|R)$ value.*

The 2-pass prediction method has an L1 loss of 0.1539 on the same set of repeating terms, which is lower than the L1 loss of variation prediction (of about 0.1753 to 0.2178), however, the features are more expensive to compute. Considering that there are only 78 repeating word pairs available in the TREC 3 dataset for training the variation prediction model, this performance is quite acceptable. The 2-pass method uses many more training samples in the range of 300 to 500, and more expensive features.

When looking at $P(t|R)$ variation prediction error together with retrieval performance, they do not always correlate with each other. For example, POS+previousP+idf+clarity has a high variation prediction error, but at the same time high MAP.

This metric divergence disappears when including association variation features (e.g. Max_assoc, Avg_assoc). The feature combination POS+previousP+Max_assoc+Avg_assoc (referred to as SPMA later on) has almost the lowest prediction error, and the highest retrieval performance. Because association variation is the major cause of $P(t|R)$ variation, the lack of such features would make the learning algorithm prone to overfitting and metric divergence is more likely to occur.

Further investigation of the POS+previousP+idf+clarity run shows that of the 90 pairs of term occurrences with the largest prediction error, 40 of them are stopwords "what", "how", "why", "is" and "done". For these words, because of their relatively low idf, a large error in $P(t|R)$ weight would not make much difference in the final retrieval score, and in turn, retrieval performance. On the other hand, because of the prevalence of those stopwords in the training set, they contribute a lot to the overall variation prediction accuracy. Simply removing stopwords in prediction accuracy evaluation solves the metric divergence problem only to some extent, because there are high $df$ terms that are not stopwords, and they can still bias the metric.

Because the performance of the SPMA feature set is the best by both variation prediction accuracy and retrieval performance, when extending this experiment on more test collections, this feature combination is selected as the best variation prediction run.

For the TREC 4 test query terms, only 13.7% (67 in a total of 489) appeared in the training set, thus, variation prediction can only make a difference for 13.7% of the test query terms. Compared to this small coverage over test terms, the impact on retrieval performance is relatively large, with the baseline (no feature) method getting an MAP of 0.1889 and the best prediction run getting an MAP of 0.1930, which is about 30% of the total retrieval performance gain of the 2-pass method. If training data covered more of the query terms, this 1-pass approach to predicting $P(t|R)$ might be as effective as the 2-pass approach. Retrieval experiments on more datasets are presented in Section 7.5.

Table 7.2: Feature correlations with $P(t|R)$ variation. Pearson/Linear correlations are computed on TREC 3 dataset with 78 repeating term pairs (instances) and TREC 4 with 250 term pairs.

| **Feature** | **TREC 3 correlation** | **# non-zero entries** | **TREC 4 correlation** | **# non-zero entries** |
|---|---|---|---|---|
| idf | 0 | 78 | 0 | 250 |
| clarity | 0 | 78 | 0 | 250 |
| dependency leaf | -0.0964 | 18 | -0.1976 | 84 |
| ADJ (POS) | 0.1509 | 6 | -0.4054 | 12 |
| NN (POS) | -0.0600 | 14 | 0.4580 | 22 |
| VB (POS) | -0.1702 | 6 | -0.2276 | 14 |
| phrase | 0.1267 | 78 | -0.0669 | 250 |
| previousP | 0.3703 | 78 | 0.4436 | 250 |
| Max_association | 0.0204 | 78 | 0.1050 | 250 |
| Avg_association | 0.1654 | 78 | 0.2160 | 250 |
| Min_associaiton | 0.2943 | 78 | 0.1799 | 250 |

To gain more insight into what features are better for $P(t|R)$ variation prediction, Table 7.2 lists feature correlation with $P(t|R)$ variation, as a measure of feature importance.

First thing to notice in Table 7.2 is that previousP and association variation features have a high coverage over the test instances (large numbers of non-zero entries), and a consistent and high correlation with the prediction target. These features are also among the most effective features from the prediction and retrieval experiments of Table 7.1.

Secondly, POS features have a low coverage over test instances, and most likely because of the low coverage, the correlations are not always consistent on the training (TREC 3) vs. the test (TREC 4) collections. The correlations seem to suggest that some of the POS features are noisy, however, for $P(t|R)$ variation prediction, what's more important is whether the POS is different, not whether it's positive or negative (e.g. a noun changing into a verb or a verb into a noun). The POS features when combined with other features lead to most effective predictions as shown in Table 7.1.

Thirdly, term specific features (idf, clarity) have a 0 linear correlation with the prediction target. This is because we use the back-forth method when generating training instances, so that the same term (same idf and clarity) always has the same number of positive and negative $P(t|R)$ differences of the same magnitude. However, a 0 linear correlation does not necessarily mean that the support vector regression algorithm cannot make use of it. Idf and clarity do affect prediction and retrieval performance in Table 7.1.

In summary, features POS+previousP+Max_assoc+Avg_assoc (SPMA) lead to one of the best $P(t|R)$ variation prediction accuracies and the best retrieval performance among the many combinations that have been tried. At the same time, these features are very efficient to compute.

## 7.5 Experiments – 1-pass Retrieval Term Weighting

This section examines the use of the efficient 1-pass $P(t|R)$ prediction designed in this chapter.

The same datasets used in Chapter 6, from 12 years of TREC ad hoc tracks (TREC 3-8) and Web tracks (TREC 9-14), are used for the evaluation in this section.

### 7.5.1 Variation Predicted $\hat{P}(t|R)$ Weighting

As shown in Section 7.3, $P(t|R)$ variation prediction can be used to predict $P(t|R)$ for terms that have appeared in the training set. For new terms that do not appear in the training set, a default term weight of 0.5 is assumed, so that the 2-pass computation can be avoided. These $P(t|R)$ predictions are then used to weight test query terms. The final relevance score is calculated according to the Relevance Model as shown in Equation 6.4, and we report the final retrieval performance. These runs are called "Variation Prediction" in Table 7.3.

Two baselines are included. The first baseline is the query likelihood language model with Dirichlet smoothing (3rd column from the left). The second baseline Variation Prediction (Average) does not use any feature in $P(t|R)$ variation prediction, and simply takes the average of the $P(t|R)$ values from the prior occurrence(s) of the term in the training set as its prediction for the same term in a test query. For reference, the 2-pass prediction performance is also listed in Table 7.3 (the rightmost column).

Table 7.3 shows that variation based prediction can be reliably used in weighting the repeating query terms in test queries. *Across all the 6 test collections with different training set sizes, the resulting retrieval performance is much better than the LM baseline, and mostly only slightly lower than that of the 2-pass method using more complex features*. In some cases, it is better. There are some cases where 1-pass prediction is 10-15% worse than the 2-pass method, mainly TREC 4 and 10. On these datasets term

Table 7.3: Mean Average Precision (MAP) of using variation predicted $\hat{P}(t|R)$ values to weight repeating terms. Non-repeating terms are assumed a 0.5 default. The Variation Prediction (Average) run simply takes the average of the historic $P(t|R)$ values of the term. The Variation Prediction (SPMA) run uses the features POS+previousP+Max_assoc+Avg_assoc, which have been found best on TREC 4. Improvements and significance levels were measured comparing the Variation Prediction (SPMA) run to the Language Model baseline. Bold faced results are significantly better than the baseline by both significance tests at $p < 0.01$.

| TREC train sets | Test set | LM baseline | Variation Prediction | | Improvement% | Term coverage | 2-pass prediction |
|---|---|---|---|---|---|---|---|
| | | | Average | SPMA | | | |
| 3 | 4 | 0.1789 | 0.1889 | 0.1930 | 7.88% | 13.7% | **0.2261** |
| 5 | 6 | 0.1586 | **0.1876** | **0.1943** | **22.51%** | 32.1% | 0.1877 |
| 3-5 | 6 | 0.1586 | **0.1887** | **0.1942** | **22.45%** | 47.6% | **0.1959** |
| 7 | 8 | 0.1923 | **0.2266** | **0.2193** | **14.04%** | 21.1% | **0.2333** |
| 3-7 | 8 | 0.1923 | **0.2288** | **0.2399** | **24.75%** | 48.4% | **0.2314** |
| 9 | 10 | 0.1627 | 0.1502 | 0.1485 | -8.73% | 18.2% | **0.1810** |
| 3-9 | 10 | 0.1627 | 0.1636 | 0.1649 | 1.35% | 52.5% | 0.1813 |
| 11 | 12 | 0.0239 | **0.0481** | **0.0471** | **96.25%** | 31.8% | **0.0597** |
| 13 | 14 | 0.1789 | 0.2149 | 0.2154 | 20.40% | 23.7% | 0.2233 |

coverage is low (2nd column from the right). In some cases where term coverage is high, e.g. TREC 8 with 3-7 for training or TREC 6, 1-pass prediction is even slightly better than the 2-pass method. The better performance confirms our hypothesis that variation prediction using historic data is easier than direct prediction of $P(t|R)$.

Test term coverage seems to correlate with retrieval performance. With more test term coverage (training on TREC 3-5 or TREC 3-7) retrieval performance is much closer to the 2-pass method. Conversely, with less term coverage, e.g. TREC 4, 10 and 14, retrieval performance is relatively lower. Especially on TREC 8 with TREC 3-7 (150 queries) for training, test term coverage increases to about 50%, and the retrieval performance of variation based prediction outperforms that of the 2-pass method. However, 50% is still a low coverage, which shows that *not all of the query terms need to be reweighted to improve retrieval*.

TREC 10 is the only dataset that Variation Prediction (SPMA) performed worse than the language model baseline, for which the method has a very low coverage over test terms. We tried to add TREC 3-8 as training data. Even though TREC 3-8 are very different (significantly smaller, and only newswire text) from the Web collection WT10g used in TREC 9 and 10, the additional training data still increases retrieval performance to slightly better than the language model baseline, although still not as good as the 2-pass method.

Theoretically, it is possible for the variation based prediction method to outperform the 2-pass prediction. In the 2-pass method, none of the 5 features exactly identify a term, thus the regression algorithm can never make use of the fact that a particular test term has appeared in the training set and that we know its true $P(t|R)$ in the training query. Given the low $P(t|R)$ variation for 62% of the term repeats, *the true $P(t|R)$ probabilities from the training queries can be closer to the true test-query $P(t|R)$ than the 2-pass predictions*.

Using features selected on TREC 4 dataset, the Variation Prediction (SPMA) run compares favorably to the Variation Prediction (Average) run which uses no features in its prediction. The performance on some collections is similar, but in 5 out of the 8 cases, using no feature hurts performance a lot. This means *feature based variation prediction is more robust than using no feature at all*.

Table 7.4: Mean Average Precision (MAP) of using variation predicted $\hat{P}(t|R)$ values to weight repeating terms. Non-repeating terms are assumed a 0.5 default, except for the Variation Prediction (hybrid) run which weights non-repeating terms using the 2-pass predictions of Chapter 6. Bold faced results are significantly better than the Language Model baseline by both significance tests at $p < 0.01$.

| TREC train sets | Test set | LM baseline | Variation Prediction | | 2-pass prediction | Variation Prediction (hybrid) |
|---|---|---|---|---|---|---|
| | | | Average | SPMA | | |
| 3 | 4 | 0.1789 | 0.1889 | 0.1930 | **0.2261** | **0.2236** |
| 5 | 6 | 0.1586 | **0.1876** | **0.1943** | 0.1877 | **0.1957** |
| 3-5 | 6 | 0.1586 | **0.1887** | **0.1942** | **0.1959** | **0.2047** |
| 7 | 8 | 0.1923 | **0.2266** | **0.2193** | **0.2333** | **0.2225** |
| 3-7 | 8 | 0.1923 | **0.2288** | **0.2399** | **0.2314** | **0.2438** |
| 9 | 10 | 0.1627 | 0.1502 | 0.1485 | **0.1810** | 0.1747 |
| 3-9 | 10 | 0.1627 | 0.1636 | 0.1649 | 0.1813 | 0.1704 |
| 11 | 12 | 0.0239 | **0.0481** | **0.0471** | **0.0597** | **0.0540** |
| 13 | 14 | 0.1789 | 0.2149 | 0.2154 | 0.2233 | 0.2227 |

To gauge the potential impact of using repeating terms in retrieval, in the rightmost column of Table 7.4, we list a hybrid run, which uses 2-pass prediction for new terms (instead of the 0.5 default) and variation based prediction (with SPMA features) for repeating terms. This Variation Prediction (hybrid) run is consistently better than the 1-pass Variation Prediction (SPMA) run, and even better than 2-pass prediction when there is over 30% test term coverage. This hybrid run is not practically helpful for retrieval efficiency, as it still needs the expensive 2-pass prediction for test queries that include a new term. The purpose of this run is to show what kind of performance is possible with larger training sets that cover more test terms.

### 7.5.2 Using Query Log to Improve Test Term Coverage

The section above shows retrieval effectiveness depends on how well the training data covers query terms that will be observed in the future. The problem now is for the query terms that have not been observed in the training set, how to better estimate $P(t|R)$ than simply assuming it to be 0.5.

#### 7.5.2.1 Bootstrapping

Because our 1-pass method produces a prediction based on prior instances of occurrence of the same term, the coverage and effectiveness of the predictions depends on how well the training data covers query terms that will be encountered in the future. Covering a large vocabulary would need to require a large number of training queries with fairly complete relevance judgments, which would be expensive or impractical to obtain.

This section provides a smaller scale study that supplements the above section and shows as proof of concept how to use a small amount of manually-labeled data and a large amount of automatically-labeled data to reduce the need for using labeled data to cover test terms.

First, a small set of queries and fairly complete relevance assessments (*manually-labeled data*) is used to train the 2-pass $P(t|R)$ prediction model of Chapter 5. This model is slow, but it is effective and can be applied to any query term.

We assume that a large query log (*unlabeled data*) is available. The query log provides good coverage of vocabulary, but does not contain relevance judgments or clickthrough information. In a dynamic envi-

ronment, the log might be updated periodically as query traffic evolves. In the experiments, we used the TREC 2007 Million Query track queries as this large query log.

Then, the 2-pass model is used to predict $P(t|R)$ for each query in the log (*automatically-labeled data*). The predictions are used as if they are the $P(t|R)$ truth to train variation prediction models. Chapter 5 and Table 7.1 suggest that these $\hat{P}(t|R)$ predictions will be close to estimates based on relevance assessments. The repeating term pairs from this automatically-labeled query log are then extracted to train the more efficient variation prediction model. During testing, the large number of query terms from the query log with predicted $\hat{P}(t|R)$ are used to predict term weights for the test terms. Thus, the slower method is used in an *offline* phase to create comprehensive training data for the more efficient method that is used *online*.

#### 7.5.2.2 Retrieval Experiments

Table 7.5: Mean Average Precision of using variation predicted $P(t|R)$ bootstrapped from TREC 2007 Million Query track queries. Words covered in TREC 13 training set still use variation predictions based on TREC 13. Those not covered by TREC 13 but covered by TREC 2007 use variation predictions based on bootstrapping. The improvement of Variation Prediction (SPMA) over the language model baseline is significant by the randomization test at $p < 0.0008$, but not significant by the sign test ($p = 0.1611$).

| TREC train sets | Test set | LM baseline | Variation Prediction | | Improvement% | Term coverage | 2-pass prediction |
|---|---|---|---|---|---|---|---|
| | | | Average | SPMA | | | |
| 13 | 14 | 0.1789 | 0.2143 | 0.2172 | 21.41% | 76.6% | 0.2233 |

We show such an experiment that tests this more ambitious approach. It uses the 50 TREC 13 description queries for training the $P(t|R)$ prediction model, the 10,000 title queries from the TREC 2007 Million Query track[1] to train the variation prediction model, and the 50 TREC 14 description queries for testing and final retrieval evaluation.

Since the basic 1-pass prediction method trained on manually labeled data (TREC 13) still performs better than the bootstrapped predictions based on automatically labeled data (TREC 2007), we use the basic 1-pass prediction for the 20% test terms covered by TREC 13. For test terms covered by the automatically labeled data but not the manually labeled data, the bootstrapping method is used. For all other test terms, a 0.5 default is used.

As seen in Table 7.5, using the bootstrapping method, term coverage increased significantly from the basic 1-pass method in Table 7.3. Of all the 316 test terms, 77% appeared in the 10,000 training queries, and 40% of the 50 test queries were fully covered. Although the improvement was not statistically significant, the retrieval performance of 0.2172 was better than that of the basic 1-pass method (0.2154 from Table 7.3). Feature based prediction (SPMA) was still better than using no feature (Variation Prediction - Average).

In Table 7.5, the sign test gives a very different significance level than the randomization test. To investigate why, we looked at the performance changes over the test queries. About 40% of the 50 queries experienced a small performance decrease. Since most of the MAP decreases are relatively small, while most of the gains are substantial compared to the baseline performance, the randomization test which considers the magnitude of the changes judges the improvement to be significant, while the sign test, which only considers the directions of the changes regardless of their magnitude, gives a different assessment.

[1]The 50 queries that come from the TREC 14 test queries were removed from the 10,000 queries, and the 50 TREC 13 description queries were added in.

## 7.6 Efficiency

Instead of the sequential steps of initial retrieval and LSI, the 1-pass $P(t|R)$ prediction approach only requires historic term information lookup and computation of the term association features, all of which are less expensive and can be done in parallel. Using an experimental system which does not exploit parallelism, on a Linux server with 2x2.8 GHz dual-core Intel Xeon processors and 8 GB of RAM, overall retrieval efficiency improved from 8 seconds per query for the 2-pass method to 3 seconds per query on half a million newswire texts, while baseline keyword retrieval took about 0.7 second. On 25 million Web documents, overall retrieval time was cut from 120 seconds to 12 seconds per query, with keyword baseline taking 9 seconds per query.

Efficiency improved 3 to 10 times over the 2-pass method. The overall efficiency of the 1-pass prediction method is getting close to keyword retrieval. Larger collections benefit more from the 1-pass method, because on the larger collections the 2-pass method requires LSI computation on more (600) top ranked documents, increasing computation complexity more than quadratically. On larger collections, the 1-pass method can also achieve efficiency closer to that of keyword retrieval. This is because the additional computation of feature generation and 1-pass prediction is amortized over the cost of retrieving and scoring a much larger set of candidate documents from a larger collection.

## 7.7 Summary

This chapter focuses on efficient methods of predicting the $P(t|R)$ probability for query terms for ad hoc retrieval. The design is based on the exploratory data analyses of Chapter 4 on how much $P(t|R)$ varies for the same term in different queries, and what causes such variation.

Efficient features have been designed to predict the variation in the $P(t|R)$ probability given two occurrences of the same term (a previous occurrence in the training set and the test occurrence). Experiments show encouraging prediction accuracy with the current features.

Across 6 different test sets, term weighting using the efficient 1-pass term recall predictions proposed in Chapter 7 leads to a performance close to the more expensive 2-pass prediction, while spending much less time. It is more appropriate for real-time response scenarios. The effectiveness of the efficient 1-pass prediction seems to depend on the coverage of the training query terms over the test terms. When test term coverage is high, retrieval performance of the variation based prediction method can be even better than that of the 2-pass method. To improve test term coverage, a new bootstrapping method is proposed, which uses both a small training set with relevance judgments and a large query log without any relevance information. The bootstrapping method significantly improves term coverage, and achieves slightly better retrieval performance than just using the small training set.

Although the focus of this chapter is on the variation of the $P(t|R)$ probability for the same term appearing in different queries, the analyses performed in this chapter can also be applied to study the $P(t|R)$ variation of a set of different terms. For example, one such question is whether terms that appear frequently in a user query log tend to have a higher $P(t|R)$ on average. Such studies would lead to new effective features for predicting $P(t|R)$.

A potential impact of this work is the process of matching the test occurrence of a term to its closest or compatible training occurrences. This matching or registration process can be naturally adapted to apply to per term expansion, or CNF style expansion. In per term expansion, a similar process exists, to discover the word sense or word use of the test query term and match the compatible and already known expansion terms to this specific occurrence of the query term. This matching process will be particularly helpful when utilizing query rewrites from large query logs to discover expansion terms, in which case, many

training queries would contain the test query term, but only the occurrences with the compatible sense and use should be used to expand the test term.

# Chapter 8

# Automatic Term Mismatch Diagnosis for Selective Query Expansion

People are seldom aware that their search queries frequently mismatch a majority of the relevant documents. This may not be a big problem for queries with a large and diverse set of relevant documents, but would largely increase the chance of search failure for less popular search needs. We aim to address the mismatch problem by developing accurate and simple queries that require minimal effort to construct. This is achieved by targeting retrieval interventions at the query terms that are likely to mismatch relevant documents.

Chapter 5 demonstrates that this term mismatch probability can be estimated reliably prior to retrieval. Typically, it is used in probabilistic retrieval models to provide query dependent term weights. This chapter develops a new use: Automatic diagnosis of term mismatch. A search engine can use the diagnosis to suggest manual query reformulation, guide interactive query expansion, guide automatic query expansion, or motivate other responses. The research described here uses the diagnosis to guide interactive query expansion, and create Boolean conjunctive normal form (CNF) structured queries that selectively expand 'problem' query terms while leaving the rest of the query untouched. Experiments with TREC Ad-hoc and Legal Track datasets demonstrate that with high quality manual expansion, this diagnostic approach can reduce user effort by 33%, and produce simple and effective structured queries that surpass their bag of word counterparts. Experiments also shows that expert created CNF expansion queries outperform baseline keyword queries by 50-300%, which underscores the importance of the term mismatch problem and CNF expansion as an effective solution. Further gains are still very likely, because expert created queries only increases the average term recall rate from 65% (unexpanded keyword queries) to 78% (fully expanded CNF queries).

## 8.1 Introduction

Vocabulary mismatch between queries and documents is known to be important for full-text search. This dissertation research formally defines the term mismatch probability, and shows (in Chapter 3) that on average a query term mismatches (fails to appear in) 40% to 50% of the documents relevant to the query. With multi-word queries, the percentage of relevant documents that match the whole query can degrade very quickly. Even when search engines do not require all query terms to appear in result documents, including a query term that is likely to mismatch relevant documents can still cause the mismatch problem: The retrieval model will penalize the relevant documents that do not contain the term, and at the same time favor documents (false positives) that happen to contain the term but are irrelevant. Since the number of

false positives is typically much larger than the number of relevant documents for a query (Greiff 1998), these false positives can appear throughout the rank list, burying the true relevant results.

This work is concerned with the term mismatch problem, a long standing problem in retrieval. What's new in this chapter is the term level diagnosis and intervention. We use automatic predictions of the term mismatch probability (Chapter 5) to proactively diagnose each query term, and to guide further interventions to directly address the problem terms. Compared to prior approaches, which typically handle the query as a whole, the targeted intervention in this work generates simple yet effective queries.

Query expansion is one of the most common methods to solve mismatch. We use the automatic term mismatch diagnosis to guide query expansion. Other forms of intervention, e.g. term removal or substitution, can also solve certain cases of mismatch, but they are not the focus of this work. We show that proper diagnosis can save expansion effort by 33%, while achieving near optimal performance.

We generate structured expansion queries of Boolean conjunctive normal form (CNF) – a conjunction of disjunctions where each disjunction typically contains a query term and its synonyms. Carefully created CNF queries are highly effective. They can limit the effects of the expansion terms to their corresponding query term, so that while fixing the mismatched terms, the expansion query is still faithful to the semantics of the original query. We show that CNF expansion leads to more stable retrieval across different levels of expansion, minimizing problems such as topic drift even with skewed expansion of part of the query. It outperforms bag of word expansion given the same set of high quality expansion terms.

## 8.2 Related Work

This section discusses how this work relates to the other research that tries to solve the mismatch problem. In particular, research on predicting term mismatch and on conjunctive normal form (CNF) structured queries forms the basis of this work.

### 8.2.1 Term Mismatch and Automatic Diagnosis

Furnas et al. (1987) were probably the first to study *vocabulary mismatch* quantitatively, by measuring how people name the same concept/activity differently. They showed that on average 80-90% of the times, two people will name the same item differently. The best term only covers about 15-35% of all the occurrences of the item, and the 3 best terms together only cover 37-67% of the cases. Even with 15 aliases, only 60-80% coverage is achieved. The authors suggested one solution to be "unlimited aliasing", which led to the Latent Semantic Analysis (LSA) (Deerwester et al. 1990) line of research.

We formally defined the *term mismatch probability* to be $P(\bar{t}|R)$, the likelihood that term $t$ does not appear in a document $d$, given that $d$ is relevant to the query ($d \in R$). Furnas et al.'s (1987) definition of vocabulary mismatch is query independent, and can be reduced to an average case of our query dependent definition.

Recent research showed that $P(t|R)$ can be reliably predicted without using relevance information of the test queries (Greiff 1998; Metzler 2008; Zhao and Callan 2010). The chapters above in this dissertation achieve the best predictions from being the first to design and use query dependent features for prediction, features such as term centrality, replaceability and abstractness.

Previously and in the chapters above, $P(t|R)$ predictions were used to adjust query term weights of inverse document frequency (idf)-based retrieval models such as Okapi BM25 and statistical language models. Term weighting is not a new technique in retrieval research, neither is predicting term weights.

This chapter is a significant departure from the prior research that predicted $P(t|R)$. We apply the $P(t|R)$ predictions in a completely new way, to automatically diagnose term mismatch problems and

inform further interventions.

### 8.2.2 CNF Structured Expansion

Query expansion is one of the most common ways to solve mismatch. In recent years, the research community has focused on expansion of the whole query, for example using pseudo relevance feedback (Lavrenko and Croft 2001). This form of expansion is simple to manage and effective. It also allows introduction of expansion terms that are related to the query as a whole, even if their relationship to any specific original query term is tenuous.

When *people* search for information, they typically develop queries in Boolean conjunctive normal form (CNF). CNF queries are used by librarians (Lancaster 1968; Harter 1986; Hensley and Hanson 1998), lawyers (Blair 2004; Baron et al. 2007; Tomlinson et al. 2008) and other expert searchers (Clarke et al. 1996; Hearst 1996; Mitra et al. 1998). Each conjunct represents a high-level concept, and each disjunct represents alternate forms of the concept. Query expansion is accomplished by adding disjuncts that cover as many ways of expressing the concept as possible. For example, the query below from TREC 2006 Legal Track (Baron et al. 2007)

*sales of tobacco to children*

is expanded manually into

(*sales* OR *sell* OR *sold*)
AND (*tobacco* OR *cigar* OR *cigarettes*)
AND (*children* OR *child* OR *teen* OR *juvenile* OR *kid* OR *adolescent*).

An expressive query language, e.g. Indri query language, could also allow combinations of phrases and Boolean operators. For example, the query "(*information* OR *text*) *retrieval*" represents two phrases *"information retrieval"* or *"text retrieval"*. Another query "*computer infected*" OR "*recover from computer virus*" represents the disjunction of two whole paraphrases. Expert users sometimes do use such complex combinations to suit their search needs and formulate general as well as compact structured queries that could match many potential relevant documents.

CNF queries ensure precision by specifying a set of concepts that must appear (AND), and improve recall by expanding alternative forms of each concept. Compared to LSA or bag of word expansion, CNF queries offer control over *what query terms to expand (the query term dimension)* and *what expansion terms to use for a query term (the expansion dimension)*.

However, these two dimensions of flexibility also make automatic formulation of CNF queries computationally challenging, and makes manual creation of CNF queries tedious. The few experiments demonstrating effective CNF expansion either used manually created queries or only worked for a special task. Hearst (1996) and Mitra et al. (1998) used ranked Boolean retrieval on manual CNF queries. Zhao and Callan (2009) automatically created CNF queries for the question answering task, based on the semantic structure of the question.

Along the two directions of term diagnosis and expansion, prior research has focused on identifying synonyms of query terms, i.e. the expansion dimension. Google has patents (Lamping and Baker 2005) using query logs to identify possible synonyms for query terms in the context of the query. Jones et al. (2006) also extracted synonyms of query terms from query logs. They called it query substitutions. Wang and Zhai (2008) mined effective query reformulations from query logs. Dang and Croft (2010) did the same with TREC Web collections. Xue et al. (2010) weighted and combined automatic whole-query reformulations, similar to the way alternative structured queries were combined by Zhao and Callan (2009). If more extensive expansions were used, the more compact CNF expansion would be a reasonable next step. Because of reasons such as suboptimal quality of expansions or insufficient number of queries for evaluation, prior research on ad hoc retrieval has not seen automatic CNF expansion to outperform key-

word retrieval. Perhaps the only exception is the related problem of context sensitive stemming (Tudhope 1996; Peng et al. 2007; Cao et al. 2008), where expansion terms are just morphological variants of the query terms, which are easier to identify and more accurate (less likely to introduce false positives).

Such prior work tried to expand any query term, and did not exploit the term diagnosis dimension, thus they essentially expanded the query terms whose synonyms are easy to find.

This work focuses on selectively expanding the query terms that really need expansion, a less well studied dimension. Exploiting this diagnosis dimension can guide further retrieval interventions such as automatic query reformulation or user interaction to the areas of the query that need help, leading to potentially more effective retrieval interventions. It also reduces the complexity of formulating CNF queries, manually or automatically. The prior research on synonym extraction is orthogonal to this work, and can be applied with term diagnosis in a real-world search system.

### 8.2.3 Simulated Interactive Expansions

Our diagnostic intervention framework is general and can be applied to both automatic and manual expansions. However, our experiments are still constrained by the availability of effective intervention methods. That is why we use manual CNF expansion, which is highly effective. To avoid using expensive and less controllable online user studies, we use existing user-created CNF queries to simulate online diagnostic expansion interactions.

Simulations of user interactions are not new in interactive retrieval experiments. Harman (1988) simulated relevance feedback experiments by using relevance judgments of the top 10 results, and evaluated feedback retrieval on the rest of the results. White et al. (2005) also used relevance judgments and assumed several user browsing strategies to simulate users' interactions with the search interface. The simulated interactions were used as user feedback to select expansion terms. Similarly, Lin and Smucker (2008) assumed a set of strategies that define how the user browses the search results, to simulate and evaluate a result browsing interface using known relevance judgments.

Compared to the prior work, our simulations never explicitly use any relevance judgments, and only make a few relatively weak assumptions about the user. We simulate based on existing user created Boolean CNF queries, which can be seen as recorded summaries of real user interactions. These fully expanded queries are used to simulate selective expansion interactions.

## 8.3 Diagnostic Intervention

This section discusses in more detail the diagnostic intervention framework, and shows how term diagnosis can be applied and evaluated in end-to-end retrieval experiments in an ideal setting.

We hope to answer the following questions. Suppose the user is willing to invest some extra time for each query, how much effort is needed to improve the initial query (in expansion effort, how many query terms need to be expanded, and how many expansion terms per query term are needed)? When is the best performance achieved? Can we direct the user to a subset of the query terms so that less effort is needed to achieve a near optimal performance? What's an effective criterion for term diagnosis?

### 8.3.1 Diagnostic Intervention Framework

The diagnostic intervention framework is designed as follows. The user issues an initial keyword query. Given the query, the system selects a subset of the query terms and asks the user to fix (e.g. expand) them. The performance after user intervention is used to compare the different diagnosis strategies.

This evaluation framework needs to control two dimensions, *the diagnosis dimension* (selecting the set of problem query terms) and *the intervention dimension* (determining the amount of intervention for each selected term). Diagnosis of terms with mismatch problems can be achieved using criteria such as low predicted $P(t|R)$ or high idf. The intervention dimension when implemented as query expansion can be controlled by asking the user to provide a certain number of expansion terms.

### 8.3.2 Query Term Diagnosis Methods

We consider two competing query diagnosis methods, idf based query term diagnosis and predicted $P(t|R)$ based diagnosis.

The idf based diagnosis selects the query terms that have the highest idf first. Idf is known to have a correlation with $P(t|R)$ (Greiff 1998) and has been used as a feature for predicting $P(t|R)$ (Greiff 1998; Metzler 2008; Zhao and Callan 2010). A rare term (high idf) usually means a higher likelihood of mismatch, while a frequent word (e.g. stopword) would have a high $P(t|R)$ probability.

Diagnosis based on predicted $\hat{P}(t|R)$ selects the query terms with the lowest $\hat{P}(t|R)$ probabilities first. This dissertation develops the best known method to predict $P(t|R)$, being the first to use query dependent features for prediction. In particular, the 2-pass prediction method of Chapter 5 uses top ranked documents from an initial retrieval to automatically extract query dependent synonyms. These synonyms are used to create some of the effective query dependent features, e.g. how often synonyms of a query term appear in top ranked documents from the initial retrieval, and how often such synonyms appear in place of the original query term in collection documents. Section 8.4 describes details of adapting the 2-pass prediction method to the datasets used in this chapter.

### 8.3.3 Possible Confounding Factors

To exactly follow this ideal framework, for each query, many user interaction experiments are needed - one experiment for each possible diagnostic intervention setup, preferably, one user per setup. Many factors need to be controlled, such as users' prior knowledge of the topic of the query, the quality of the manual interventions, users' interaction time and interaction method (whether retrieval results are examined), so that the final retrieval performance will reflect the effect of the diagnostic component instead of random variation in the experiment setup. These factors are difficult to eliminate even with hundreds of experiments per query. We show how simulations may help in the section below.

## 8.4 Experimental Methodology

In this section, we design the retrieval experiments to evaluate diagnostic interventions. We explain how user simulations may be an appropriate substitute for costly online experiments with human subjects. We explain how to design the diagnostic intervention experiment so that it measures the effects of term diagnosis, minimizing effects from confounding factors such as the quality of the post-diagnosis interventions. We examine the datasets used for this simulation, in particular how well the manual CNF queries fit the simulation assumptions. We also describe the whole evaluation procedure, including the implementation of the mismatch diagnosis methods which (together with the user intervention) produce the post-intervention queries, the retrieval model behind query execution, and the evaluation metrics used to measure retrieval effectiveness.

We focus on *interactive expansion* as the intervention method, using existing CNF queries to simulate interactive expansion. This is due to both the effectiveness of query expansion to solve mismatch and the lack of user data for other types of interventions.

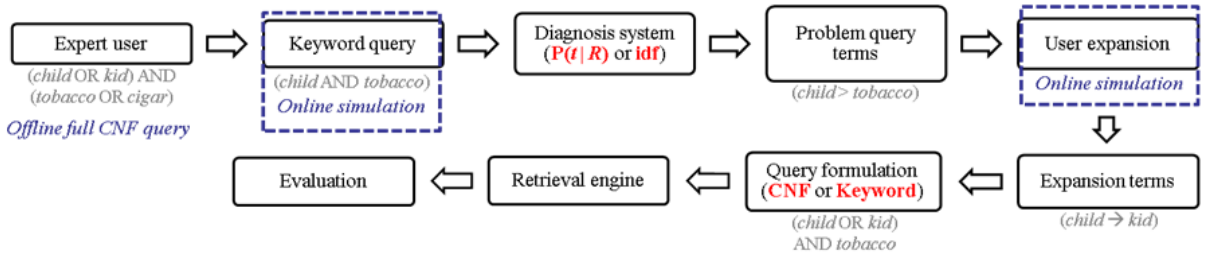



Figure 8.1: Simulated diagnostic expansion, with query examples in gray, simulated steps in dashed boxes and methods to test in bold red font.

### 8.4.1 Simulated User Interactions

As explained in Section 8.3.3, instead of online user studies, we use offline automatic simulations to evaluate the diagnostic intervention methods, as sketched in Figure 8.1. There are three players in the simulation, the *user*, the *diagnostic expansion retrieval system* and the *simulation based evaluation system*. Before evaluation, the user creates fully expanded CNF queries. These manually created CNF queries are used by the evaluation system to simulate selective expansions, and are accessible to the diagnostic retrieval system only through the simulation system. During evaluation, the simulation system first extracts a basic no-expansion keyword query from the CNF query, feeding the keyword query to the diagnosis system. The diagnostic expansion system automatically diagnoses which keywords are more problematic. Then, the simulation system takes the top problem term(s), and extracts a certain number of expansion terms for each selected query term from its corresponding conjunct in the manual CNF query. This step simulates a user expanding the selected query terms. The number of problem query terms to expand and the number of expansion terms to include are controlled by the simulation system, which will evaluate retrieval at five different selection levels and five different expansion levels. Finally, given these expansion terms for each selected query term, the diagnostic expansion system forms an expansion query and does retrieval.

For example, based on the CNF query in Section 8.2.2, the diagnosis method is given the keyword query *sales tobacco children*. It may see *children* as more problematic than the other terms, then a full expansion of this problem term would produce the query *sales* AND *tobacco* AND (*children* OR *child* OR *teen* OR *juvenile* OR *kid* OR *adolescent*), whose retrieval performance is evaluated as the end result of diagnostic expansion. If the evaluation system selects two query terms *sales* and *children* for expansion, with a maximum of one expansion term each, the final query would be (*sales* OR *sell*) AND *tobacco* AND (*children* OR *child*). These diagnostic expansion queries are partial expansions simulated using the fully expanded queries created by real users.

Our simulation allows us to answer the same set of questions about the diagnostic expansion system which we hope to answer through online user interactions, and requires simpler experiments. In our simulations, the same set of expansion terms is always used for a given query term, those from its corresponding CNF conjunct. Doing so minimizes the variation from the expansion terms as we measure the effects of the diagnosis component. The order in which expansion terms are added for a query term is also fixed, in the same order as they appear in the CNF conjunct. This way, we can tweak the level of expansion by gradually including more expansion terms from the lists of expansion terms, and answer how much expansion is needed for optimal performance.

Our simulation makes three assumptions about the user expansion process. We examine them below.

*Expansion term independence assumption*: Expansion terms from fully expanded queries are held

back from the query to simulate the selective and partial expansion of query terms. This simulation is based on the assumption that the user (a random process that generates expansion terms) will produce the same set of expansion terms for a query term whenever asked to expand any subset of the query terms. Equivalently, given the query, the expansion process for one query term does not depend on the expansion of other query terms. In reality, a human user focusing on a subset of the query terms can typically achieve higher quality expansion. Thus, selective expansion may actually do better than the reported performance from the simulations.

*Expansion term sequence assumption*: Controlling to include only the first few expansion terms of a query term simulates and measures a user's expansion effort for that query term. It is assumed that the user would come up with the same sequence of expansion terms for each query term, no matter whether the user is asked to expand a subset or all of the query terms. A downside of this simulation is that we do not know exactly how much time and effort the user has spent on each expansion term.

*CNF keyword-query induction assumption*: Instead of actually asking users to expand their initial queries, preexisting fully expanded CNF style queries are used to infer the original keyword query and to simulate the expansion process. For example, given the CNF query in Section 8.2.2, the original keyword query is assumed to be (*sales tobacco children*), and each conjunct includes the expansion terms for each query term, e.g. *sales* is expanded with *sell* and *sold*. This simulation assumes that the original keyword query can be reconstructed from the manual CNF query, which could be missing some original query terms (*of* and *to* in the example) or introduce new query terms into the original keyword query issued by the user. However, as long as we use highly effective CNF queries, it is safe to use the CNF induced keyword queries as the no-expansion baseline.

We also made an effort to ensure that our 'reverse-engineered' keyword query is faithful to the vocabulary of the original query. Given the TREC official description query, we try to use the first term from each conjunct that appears in this description to reconstruct the keyword query. For conjuncts that do not have description terms, the first term in the conjunct is used.

For example, the query described as *sales of tobacco to children*, with CNF query (*sales* OR *sell* OR *sold*) AND (*tobacco* OR *cigar* OR *cigarettes*) AND (*children* OR *child* OR *teen* OR *juvenile* OR *kid* OR *adolescent*), would have (*sales tobacco children*) as the unexpanded keyword query. If the description were *sell tobacco to children*, the keyword query would be instead (*sell tobacco children*), even when *sales* appears first in its conjunct.

### 8.4.2 Effects of Confounding Factors

Using user simulations instead of real users can eliminate confounding factors such as the user's prior knowledge of the topic of the query and other details of the user interaction process.

This work tests the hypothesis that term diagnosis can effectively guide query expansion. However, two factors directly determine the end performance of diagnostic expansion, 1) the effectiveness of term diagnosis, and 2) the benefit from expansion.

Since our focus is on diagnosis, not query expansion, one of the most important confounding factors is the quality of the expansion terms, which we leave out of the evaluation by using a fixed set of high quality expansion terms from manual CNF queries to simulate an expert user doing manual expansion.

Automatic query expansion is more desirable in a deployed system, but the uncertain quality of the expansion terms can confuse the evaluation. Thus, it is not considered in this dissertation.

### 8.4.3 Datasets and Manual CNF Queries

Datasets with high quality manual CNF queries are selected to simulate and evaluate diagnostic expansion. Four different datasets have been used, those from TREC 2006 and 2007 Legal tracks, and those from TREC 3 and 4 Ad hoc tracks. They are selected in pairs, because training data is needed to train the $P(t|R)$ prediction model. Here, the TREC 2006 (39 queries) and TREC 3 (50 queries) datasets are used for training the baseline model parameters and the $P(t|R)$ prediction models, while TREC 2007 (43 queries) and TREC 4 (50 queries) are used for testing.

#### 8.4.3.1 TREC Legal Track Datasets

The TREC Legal tracks contain Boolean CNF queries created through expert user interaction. They are fairly specific, averaging 3 conjuncts per query, i.e., 3 concepts conjoined to form a query. The information needs of the Legal track queries are fictional, but mimic the real cases.

The lawyers who created the TREC Legal queries know what the collection is, and have expert knowledge of what terminology the corpus documents might use to refer to a concept being requested. The lawyers would give very high priority to the recall of the queries they create. They tried to fully expand every query term, so as not to miss any potentially relevant document. An effort to avoid over-generalizing the query was also made. However, the lawyers never looked at the retrieval results when creating these CNF style queries. We call this a case of *blind user interaction*, because no corpus information is accessed during user interaction. We use the Boolean queries from Zhu et al. (2008), which achieved near best performance in TREC 2007 Legal track.

The 2006 and 2007 TREC Legal tracks share the same collection of documents. These are about 6.7 million tobacco company documents made public through litigation. They are on average 1362 words long. Many of them are OCR texts, and contain spelling and spacing errors.

For relevance judgments, because of the relatively large size of the collection, a sampled pooling strategy was adopted in Legal 2007, with 555.7 judgments per query and 101 judged relevant documents per query.

More details about the dataset, the information needs, query formulation procedure, and relevance judgments can be found in the overview papers (Baron et al. 2007; Tomlinson et al. 2008).

#### 8.4.3.2 TREC Ad hoc Track Datasets

For the TREC 3 and 4 Ad hoc track datasets, high quality manual CNF queries were created by the University of Waterloo group (Clarke et al. 1996). An example query is

(*responsibility* OR *standard* OR *train* OR *monitoring* OR *quality*)
AND (*children* OR *child* OR *baby* OR *infant*)
AND "*au pair*"

where the "*au pair*" is a phrase.

The information needs for the TREC 3 and 4 Ad hoc tracks are simpler (or broader), averaging 2 conjuncts per query.

The Waterloo queries were created for the MultiText system by an Iterative Searching and Judging (ISJ) process (Clarke et al. 1996). These queries were manually formulated with access to the results returned by the retrieval system, thesaurus and other external resources of knowledge. This constitutes a case of *user* and *corpus interaction*. Quality of the manual Boolean queries is ensured by examining the retrieval results, thus, should be better than those created from blind user interaction of the Legal tracks. Since the interaction with search results, expansion processes of the query terms may not be independent of each other. For example, in order to discover the expansion term of a query term, one may need to

expand another query term first, to bring up a result document that contains the expansion term. Thus, the expansion independence assumption (of Section 8.4.1) is more likely to be violated by the ISJ queries than by the Legal ones.

The TREC 3 and 4 Ad hoc tracks used different collections, but they both consisted of newswire texts published before 1995. Each collection has about 0.56 million documents. The texts are non-OCR, thus cleaner than the Legal documents.

The relevance judgments of the Ad hoc tracks are deeper, because the collections are much smaller. The TREC 4 Ad hoc track made 1741 judgments per query with 130 relevant. More details are provided by Harman (1995, 1996).

For all documents and queries, the Krovetz stemmer was used (more conservative than Porter), and no stopwords were removed.

### 8.4.4 Term Diagnosis Implementation

We explain the implementation of the diagnosis methods, idf and predicted $P(t|R)$, in more detail.

Idf is calculated as $\log((N-df)/df)$, where $N$ is the total number of documents in the collection and $df$ is the document frequency of the term. This follows the RSJ formulation (Robertson and Spärck Jones 1976).

For $P(t|R)$ prediction, we closely follow the 2-pass $P(t|R)$ prediction method of Chapter 5. [1] Automatic features used for prediction include idf, and the three features derived from applying latent semantic analysis (LSA) (Deerwester et al. 1990) over the top ranked documents of an initial keyword retrieval. For training purposes, $P(t|R)$ truth is calculated as $(r+1)/(|R|+2)$, where $r$ is the number of relevant documents containing $t$ and $|R|$ the total number of relevant documents for the query, with Laplace smoothing used. Support Vector Regression with RBF kernel is used to learn the prediction model.

There are 3 parameters: The number of top ranked documents for LSA, which is set at 180 for the Ad hoc datasets and 200 for the Legal track datasets, based on a monotonic relationship between this parameter and the total number of collection documents observed in Chapter 6. The number of latent dimensions to keep is fixed at 150, and the gamma parameter which controls the width of the RBF kernel is fixed at 1.5 (as in Chapter 6).

Chapter 6 also used a feature that indicated whether a word in the query appears as a leaf node in the dependency parse of the query. Here, the feature is assumed to be 0 for all query terms, because the unexpanded query is usually not a natural language sentence or phrase, hence parsing may be inappropriate.

A small number of the original terms in these CNF queries are phrases, windowed occurrences or other complex structures. They are assumed to have a $P(t|R)$ value of 0.5. The LSA component of the Lemur Toolkit is not designed to handle these complex terms, preventing the use of Chapter 6's model. This is a small handicap to our $P(t|R)$ prediction implementation, but not to the idf method, which is based on accurate df values calculated by the Indri search engine.

### 8.4.5 The Retrieval Model

To achieve a state-of-the-art performance, the retrieval model needs to rank collection documents using the Boolean CNF queries. Before the mid 1990's, unranked Boolean was popular. Later research found ranked keyword to be more effective. However, to be fair, a *ranked* Boolean (e.g. soft or probabilistic) model should be used to compare with other ranking approaches.

[1] The more efficient 1-pass prediction of Chapter 7 can also be used to provide the $P(t|R)$ predictions needed in this chapter. But the focus of this chapter is to use the predictions in diagnostic expansion, instead of comparing the two prediction methods.

This chapter adopts the language model framework, using probabilistic Boolean query execution (with Lemur/Indri version 4.10) (Metzler and Croft 2004). The Boolean OR operator is still the hard OR, treating all the synonyms as if they are the same term for counting term- and document-frequencies (i.e. $\#syn$ operator in Indri query language). The Boolean AND operator is implemented as the probabilistic AND (the Indri $\#combine$ operator) to produce a ranking score.

Equations 8.1 and 8.2 show how the retrieval model scores document $d$ with query $(a$ OR $b)$ AND $(c$ OR $e)$, or equivalently $\#combine(\#syn(a\ b)\ \#syn(c\ e)$ in Indri query language. $tf(a,d)$ is the number of times term $a$ appears in document $d$. $\mu$ is the parameter for Dirichlet smoothing, which is set at 900 for the Ad hoc datasets and 1000 for the Legal datasets based on training.

$$\begin{aligned} & Score((a \text{ OR } b) \text{ AND } (c \text{ OR } e), d) \\ = \quad & P((a \text{ OR } b) \text{ AND } (c \text{ OR } e)|d) \\ = \quad & P((a \text{ OR } b)|d) * P((c \text{ OR } e)|d) \end{aligned} \tag{8.1}$$

$$\begin{aligned} & P(a \text{ OR } b)|d) \\ = \quad & (tf(a,d) + tf(b,d) + \mu * (P(a|C) + P(b|C)))/(length(d) + \mu) \\ = \quad & P(a|d) + P(b|d) \qquad \text{(under Dirichlet smoothing)} \end{aligned} \tag{8.2}$$

This language model based ranked Boolean model is not the only possibility. Other ranked Boolean models include using the Boolean query as a two-tiered filter for the keyword rank list (Hearst 1996; Zhao and Callan 2009), or using the Okapi BM25 model for the conjunction (Tomlinson 2007), or using probabilistic OR for the expansion terms (in Indri query language, $\#or$ instead of $\#syn$), or using the p-norm Boolean ranking model (Salton et al. 1983). We have tried some basic variations of the language model ranked Boolean model. Our pilot study shows that for our datasets, tiered filtering is sometimes worse than probabilistic Boolean, mostly because of the inferior ranking of the keyword queries. Probabilistic OR ($\#or$) is numerically similar to treating all expansion terms the same as the original term ($\#syn$), and the two methods perform similarly in retrieval. We did not try Okapi or p-norm, because the focus of this chapter is $P(t|R)$ based diagnostic expansion, not to find the best ranked Boolean model. What is needed is one ranked Boolean model that works.

### 8.4.6 Evaluation Measures

We use standard TREC evaluation measures for the datasets. Traditionally, pooled judgments and precision at certain cutoffs have been used in TREC. *Mean Average Precision (MAP)* at top 1000 is a summary statistic that cares about both top precision and precision at high recall levels, and has been used as the standard measure in TREC Ad hoc and Legal tracks.

The *statAP* measure (Aslam and Pavlu 2007) is the standard measure for TREC Legal 2007. StatAP is an unbiased statistical estimate of MAP designed to work with sampled pooling. It is unbiased in the sense that if all pooled documents were judged, the MAP value would have been the same as the mean of the estimated statAP. In traditional TREC pooling, the top 50 to top 100 documents from each submitted rank list are pooled, and all pooled documents are judged. In sampled pooling, only a sampled subset of the pool is judged. The idea is to use importance sampling to judge fewer documents while maintaining a reliable estimate of MAP. Highly ranked documents from multiple pooled submissions are more likely to be relevant, and they are sampled more by importance sampling. StatAP takes into account these sampling

probabilities of the judged relevant documents, so that during evaluation, a judged relevant document with sampling probability $p$ would be counted as a total of $1/p$ relevant documents. This is because on average $1/p - 1$ relevant documents are missed during the sampling procedure, and they are being represented by that one sampled relevant document.

For queries where some relevant documents have low sampling probabilities, statAP estimates can deviate from the true AP a lot, but according to Tomlinson et al. (2008), when averaged over more than 20 queries, statAP provides a reliable estimate.

## 8.5 Experiments

These experiments test two main hypotheses. *H1*: Mismatch diagnosis can direct expansion to the query terms that need expansion. *H2*: Boolean CNF expansion is more effective than bag of word expansion with the same set of high quality expansion terms. To test H1, the *first* experiment verifies the accuracy of idf and $P(t|R)$-prediction based term diagnosis against the true $P(t|R)$. The *second* experiment shows the effects of diagnosis by evaluating overall retrieval performance along the query term dimension (5 diagnostic selection levels) and the expansion dimension (5 expansion levels). The *third* experiment compares predicted $P(t|R)$ diagnosis with idf based diagnosis. H2 is tested by the *fourth* experiment comparing CNF and bag-of-word expansion at various levels of diagnostic expansion.

### 8.5.1 Baseline – No Expansion

Listed below is the retrieval performance of the no expansion keyword retrieval baseline on the two test sets.

Table 8.1: Performance of the baseline no-expansion run.

| **Dataset** | Legal 2007 (MAP/statAP) | TREC 4 (MAP) |
|---|---|---|
| **no expansion** | 0.0663/0.0160 | 0.1973 |

### 8.5.2 Mismatch Diagnosis Accuracy

Our goal is to use $\hat{P}(t|R)$ predictions to diagnose the query terms, to rank them in a priority order for the user to fix (expand). This section is a unit test of the diagnosis component, in which accuracy is measured by how well the diagnosis method identifies the most problematic query terms (those most likely to mismatch). We measure how well the priority order (e.g. ascending predicted $P(t|R)$) ranks the query term with the true lowest $P(t|R)$, thus use Mean Reciprocal Rank (MRR) as the measure. Rank correlation measures do not distinguish ranking differences at the top vs. bottom of the rank lists, thus are less appropriate here.

On the Legal 2007 dataset, predicted $P(t|R)$ achieves an MRR of 0.6850, significantly higher than the MRR of 0.5523 of the idf method, significant at $p < 0.026$ by the two tailed t-test. The idf method is still much better than random chance which has an MRR of 0.383, given the average 3 conjuncts per query.

This result that $P(t|R)$ prediction using Chapter 6's method is better than idf, and idf is better than random is consistent with prior research that predicted $P(t|R)$ – (Greiff 1998) and Chapter 6.

### 8.5.3 Diagnostic Expansion Retrieval Results

Table 8.2: Retrieval performance (measured by *MAP*) of the **two selective CNF expansion methods** on TREC 2007 Legal track, by selecting $n$ query terms for expansion and expanding each of the selected query terms with at most $m$ manual expansion terms. The baseline unexpanded queries produced an MAP of 0.0663. * means significantly better than the no expansion baseline by both randomization & sign tests at $p < 0.05$. **: $p < 0.01$ by both tests, ***: $p < 0.001$ by both, #: $p < 0.0001$ by both tests. (Same notation is used for the other tables in this chapter.)

| $\backslash\, n$ | 1 | | 2 | | 3 | | 4 | | All |
|---|---|---|---|---|---|---|---|---|---|
| $m\backslash$ | idf | $\hat{P}(t\|R)$ | idf | $\hat{P}(t\|R)$ | idf | $\hat{P}(t\|R)$ | idf | $\hat{P}(t\|R)$ | (same) |
| 1 | 0.0722 | 0.0778* | 0.0802* | 0.0825** | 0.0892** | 0.0896*** | 0.0893** | 0.0904*** | 0.0901*** |
| 2 | 0.0780* | 0.0805** | 0.0825* | 0.0921*** | 0.0916*** | 0.0938# | 0.0947*** | 0.0961# | 0.0971# |
| 3 | 0.0766 | 0.0806** | 0.0844** | 0.0927*** | 0.0938*** | 0.0965# | 0.0969*** | 0.0988# | 0.0997# |
| 4 | 0.0770 | 0.0809** | 0.0859** | 0.0948# | 0.0968# | 0.0993# | 0.0996# | 0.1015# | 0.1024# |
| All | 0.0754 | 0.0798* | 0.0862** | 0.0958*** | 0.0986# | 0.1008# | 0.1016# | 0.1031# | 0.1039# |

Table 8.3: Retrieval performance (measured by *statAP*) of the **two selective CNF expansion methods** on TREC 2007 Legal track, by selecting $n$ query terms for expansion and expanding each of the selected query terms with at most $m$ manual expansion terms. The baseline unexpanded queries produced a statAP of 0.0160. (Statistical significance tests are omitted, as they are inappropriate for the sampling based statAP measure, which can be unstable on individual queries (Tomlinson et al. 2008).)

| $\backslash\, n$ | 1 | | 2 | | 3 | | 4 | | All |
|---|---|---|---|---|---|---|---|---|---|
| $m\backslash$ | idf | $\hat{P}(t\|R)$ | idf | $\hat{P}(t\|R)$ | idf | $\hat{P}(t\|R)$ | idf | $\hat{P}(t\|R)$ | (same) |
| 1 | 0.0164 | 0.0233 | 0.0184 | 0.0246 | 0.0279 | 0.0282 | 0.0279 | 0.0282 | 0.0282 |
| 2 | 0.0176 | 0.0255 | 0.0189 | 0.0289 | 0.0273 | 0.0295 | 0.0288 | 0.0302 | 0.0300 |
| 3 | 0.0175 | 0.0256 | 0.0191 | 0.0290 | 0.0280 | 0.0304 | 0.0291 | 0.0307 | 0.0304 |
| 4 | 0.0176 | 0.0256 | 0.0194 | 0.0295 | 0.0292 | 0.0311 | 0.0301 | 0.0317 | 0.0314 |
| All | 0.0185 | 0.0345 | 0.0381 | 0.0490 | 0.0491 | 0.0504 | 0.0493 | 0.0508 | 0.0505 |

Tables 8.2, 8.3 and 8.4 report the expansion retrieval performance of predicted-$P(t|R)$ based and idf based diagnostic expansion, following the evaluation procedure detailed in Section 8.4.1. The results are arranged along two dimensions of user effort, the number of query terms selected for expansion, and the maximum number of expansion terms to include for a selected query term.

For example, results reported in column 2 row 2 selects 1 original query term of the highest idf for expansion, and a maximum of 1 expansion term is included for the selected query term. When the manual CNF query doesn't expand the selected query term, no expansion term will be included in the final query.

We are most concerned with the performance changes along each row of the tables, which are caused by the diagnosis methods. In Figure 8.2, we compare the relative performance gains of the different diagnosis methods as more query terms are being selected for expansion. Results based on the last row of Tables 8.3 and 8.4 are presented in Figure 8.2. No expansion is 0%, and full expansion of all query terms gets 100%. With only 2 query terms selected for expansion, predicted $\hat{P}(t|R)$ diagnosis is achieving 95% or 90% of the total gains of CNF expansion. Idf diagnosis is only achieving 64% or 83% of the total gains with 2 query terms, and need to fully expand 3 query terms to reach a performance close to the best (full expansion of all query terms). Thus, *predicted $\hat{P}(t|R)$ based diagnosis saves 1/3 of users' expansion effort while still achieving near optimal retrieval performance*.

Table 8.4: Retrieval performance (measured by *MAP*) of the **two selective CNF expansion methods** on TREC 4 Ad hoc track, by selecting $n$ query terms for expansion and expanding each of the selected query terms with at most $m$ manual expansion terms. The baseline unexpanded queries produced an MAP of 0.1973.

| $\backslash\, n$ | 1 | | 2 | | 3 | | 4 | | All |
|---|---|---|---|---|---|---|---|---|---|
| $m\backslash$ | idf | $\hat{P}(t\|R)$ | idf | $\hat{P}(t\|R)$ | idf | $\hat{P}(t\|R)$ | idf | $\hat{P}(t\|R)$ | (same) |
| 1 | 0.2087 | 0.2279*** | 0.2341* | 0.2366** | 0.2350** | 0.2358** | 0.2356** | 0.2358** | 0.2358** |
| 2 | 0.2135 | 0.2392# | 0.2503** | 0.2541*** | 0.2552*** | 0.2567*** | 0.2578*** | 0.2581*** | 0.2581*** |
| 3 | 0.2187* | 0.2435*** | 0.2538*** | 0.2539# | 0.2589*** | 0.2608# | 0.2619# | 0.2622# | 0.2622# |
| 4 | 0.2242* | 0.2489# | 0.2654*** | 0.2659# | 0.2706*** | 0.2731# | 0.2753# | 0.2756# | 0.2756# |
| All | 0.2319** | 0.2526*** | 0.2775# | 0.2835# | 0.2875*** | 0.2916# | 0.2935# | 0.2938# | 0.2938# |

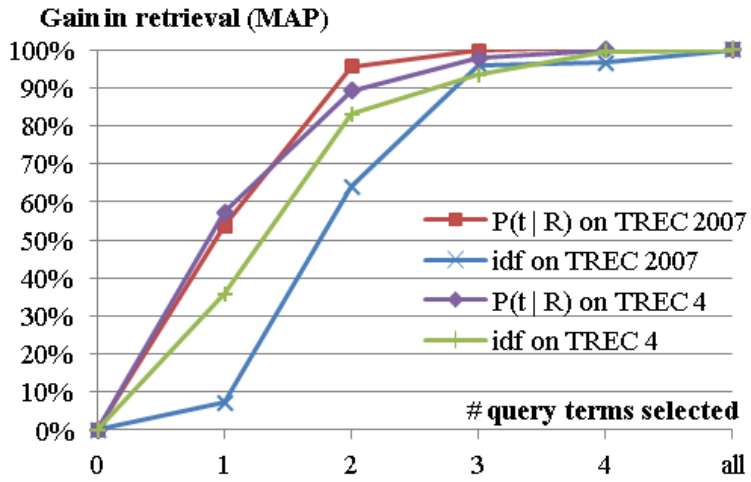



Figure 8.2: Relative retrieval performance gains of diagnostic expansion as the number of query terms selected for expansion increases. Calculated based on the last row of Tables 8.3 and 8.4.

Tables 8.2, 8.3 and 8.4 show that the more query terms selected for expansion, the better the performance. This is not surprising, as we are using carefully expanded manual queries. Similarly, including more expansion terms (along each column) almost always improves retrieval, except for the idf method in Table 8.2 with only one query term selected for expansion.

The improvement over the no expansion baseline becomes significant after expanding two query terms for the idf method, and after only expanding one query term for predicted $P(t|R)$. The retrieval accuracy gains of full expansion over no expansion is around 50-300%, which underscores the importance of the term mismatch problem, and CNF expansion as an effective solution.

Overall, $P(t|R)$ diagnostic expansion is more stable than the idf method. This shows up in several areas. 1) Including more expansion terms always improves performance, even when only one original query term is selected for expansion. 2) Performance improvement over the no expansion baseline is significant even when only including one expansion term for one query term. These are not true for idf diagnosis. 3) Only two query terms need to be selected for expansion to achieve a performance close to the best, 33% less user effort than that of idf based diagnosis.

The statAP measure from Table 8.3 correlates with the MAP measure, however, the sudden increases in statAP from the 2nd last row to the last row are not present in the case of MAP. A bit of investigation

shows that 1 to 2 queries benefited a lot from the extra (more than 4) expansion terms. The benefit is because of the successful matching of some relevant documents with low sampling probability, which increases statAP a lot, but not MAP. On the Legal 2007 dataset, the query that benefited most from the more than 4 expansion terms is about James Bond movies. Certainly, there are more than 4 popular James Bond movies.

Overall, predicted $P(t|R)$ can effectively guide user expansion to improve retrieval. Expanding the first few query terms can result in significant gains in retrieval. The gain diminishes as more query terms are expanded, eventually leading to the best performance of expanding every query term.

#### 8.5.3.1 $\hat{P}(t|R)$ vs. Idf Based Diagnostic Expansion

The subsection above shows that diagnosis can help reduce expansion effort, and that $\hat{P}(t|R)$ diagnosis results in more stable retrieval than idf. This section directly compares two retrieval experiments using $\hat{P}(t|R)$ vs. idf guided expansion.

The two experiments both select 1 query term for full expansion (Table 8.2 last row, 2nd vs. 3rd column from the left). The MAP difference between 0.0754 of idf and 0.0798 of $\hat{P}(t|R)$ is not statistically significant. We investigate why below. According to the diagnostic expansion framework, two causes are possible, 1) the $\hat{P}(t|R)$ predictions are poor, selecting the wrong query terms to expand, or 2) the quality of the expansion terms that idf selected happen to be higher, causing the idf method to have better MAP sometimes, thus decreasing statistical significance.

To separate the effects of diagnosis and expansion, we plot the Legal 2007 queries along two dimensions in the scatter-plot Figure 8.3. The $x$ axis represents the diagnosis dimension: the difference between the true $P(t|R)$ of the two query terms selected by lowest predicted $\hat{P}(t|R)$ and highest idf. The $y$ axis represents the expansion performance dimension: the difference between the Average Precision (AP) values of the two methods on a query. When idf and predicted $\hat{P}(t|R)$ happen to select the same term to expand for a given query, that query would be plotted on the origin ($x = 0$, $y = 0$) – no difference in both diagnosis and expansion.

From Figure 8.3, firstly, most points have $x < 0$, meaning the $\hat{P}(t|R)$ predictions are better at finding the low $P(t|R)$ terms than the idf based diagnosis. Secondly, most points are in the top left and bottom right quadrants, supporting the theory that expanding the term with lower $P(t|R)$ leads to better retrieval performance. In the bottom right quadrant, occasionally idf method picks the right terms with lower $P(t|R)$ to expand, and does better in retrieval.

However, there are three outliers in the bottom left quadrant, which are not fully explained by our theory. At the bottom left quadrant, predicted $\hat{P}(t|R)$ does identify the right term with a lower $P(t|R)$, but the retrieval performance is actually worse than that of idf guided expansion.

By looking into these three queries, we found that the manual queries number 76 and 86 do not have any expansion terms for the query terms selected by $\hat{P}(t|R)$, while the idf selected terms do have effective expansion terms. All such queries where a query term without expansion terms is selected are annotated with diamond shaped borders in the plot. Query number 55 is because of poor expansion term quality. The $\hat{P}(t|R)$ method selects the chemical name *apatite* for expansion, which represent a class of chemicals. The manual expansion terms seem very reasonable, and are just names or chemical formulas of the chemicals belonging to the *apatite* family. However, the query is really just about *apatite rocks* as they appear in nature, not any specific chemical in the apatite family. Thus, even expansion terms proposed by experts can still sometimes introduce false positives into the rank list, and this problem cannot be easily identified without corpus interaction, e.g. examining the result rank list of documents.

If these 3 queries were removed from evaluation, predicted $\hat{P}(t|R)$ guided expansion would be significantly better than idf guided expansion, at $p < 0.05$ by the two tailed sign test.

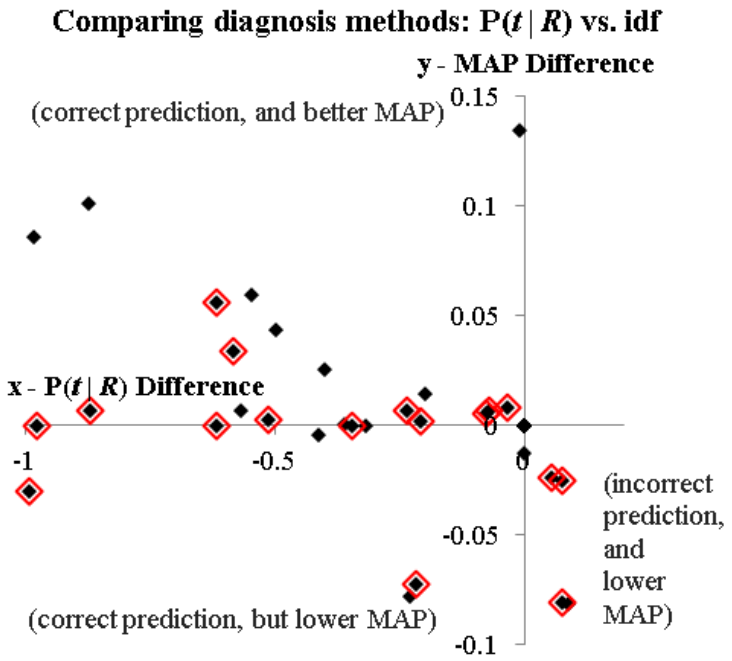



Figure 8.3: Difference in prediction accuracy vs. difference in MAP for the two selective query expansion methods on 43 TREC 2007 Legal Track queries. $X$ axis shows the difference in true $P(t|R)$ between the first query terms selected by each method. $Y$ axis shows the difference in MAP between queries expanded by each method. The differences are calculated as that from predicted $\hat{P}(t|R)$ based diagnosis minus that from idf based diagnosis. Points that are surrounded by a diamond represent queries in which one method selected a term that had no expansions.

Of the 50 TREC 4 queries, similarly, 4 queries are outliers. In two cases, the $\hat{P}(t|R)$ selected query terms do not have manual expansion terms. In one query, $\hat{P}(t|R)$ prediction did not select the right term, but MAP is higher than idf diagnosis, because the idf selected query term has poor expansion terms. In one query, the retrieval performance does not differ at top ranks, and the idf method only excels after 30 documents.

Overall, most queries confirm our hypothesis that expanding the query term most likely to mismatch relevant documents leads to better retrieval, and better diagnosis also leads to better retrieval.

This analysis also shows the inherent difficulty of evaluating term diagnosis in end-to-end retrieval experiments. Even with high quality manual expansion terms, there is still some variation in the quality of the expansion interventions, which can still interfere with the assessment of the diagnosis component.

### 8.5.4 Boolean CNF vs. Bag of Word Expansion

We compare CNF style expansion with *two* advanced bag-of-word expansion methods.

For a fair comparison with manual CNF expansion, our *first* bag of word expansion baseline also uses the set of manual expansion terms selected by predicted $\hat{P}(t|R)$. Expansion terms are then grouped and combined with the original query for retrieval.

To make this baseline strong, both individual expansion terms and the expansion term set can be weighted. The individual expansion terms are weighted with the Relevance Model weights (Lavrenko and

Table 8.5: Retrieval performance (measured by *MAP/statAP*) of $\hat{P}(t|R)$ **guided bag of word expansion** on TREC Legal track 2007, by selecting $n$ query terms for expansion and expanding each of the selected query terms with at most $m$ manual expansion terms. The baseline unexpanded queries produced an MAP/statAP of 0.0663/0.0160. Reported are the better performance of the uniform weighting and Relevance Model weighting runs.

| $\backslash\, n$ <br> $m\backslash$ | 1 | 2 | 3 | 4 | All |
|---|---|---|---|---|---|
| 1 | 0.0755** /0.0210 | 0.0744 /0.0172 | 0.0808 /0.0271 | 0.0768 /0.0260 | 0.0764***/0.0260 |
| 2 | 0.0795** /0.0229 | 0.0814** /0.0216 | 0.0892** /0.0242 | 0.0883** /0.0243 | 0.0867***/0.0242 |
| 3 | 0.0789** /0.0217 | 0.0829***/0.0221 | 0.0880** /0.0206 | 0.0894** /0.0207 | 0.0878** /0.0206 |
| 4 | 0.0789** /0.0217 | 0.0821** /0.0205 | 0.0908***/0.0203 | 0.0993***/0.0213 | 0.0927***/0.0214 |
| All | 0.0791** /0.0219 | 0.0833** /0.0217 | 0.1006# /0.0207 | 0.1038# /0.0211 | 0.1014# /0.0200 |

Croft 2001) from an initial keyword retrieval, with the parameter (the number of feedback documents) tuned on the training set. Manual expansion terms that do not appear in the feedback documents are still included in the final query, but a minimum weight is used to conform to the relevance model weights. Uniform weighting of the expansion terms was also tried. It is more effective than relevance model weights when expansion is more balanced, i.e. more than 3 query terms are selected for expansion. When combining the expansion terms with the original query, the combination weights are 2-fold cross-validated on the test set.

Table 8.5 shows the *best case* of both relevance-model-weight and uniform-weight bag of word expansion. Bag of word expansion performs worse than CNF expansion in almost all the different setups. The best performance is achieved with full expansion of 4 query terms, with a MAP of 0.1038, slightly lower than that of CNF (0.1039 in Table 8.2), however, the statAP value of 0.0211 is much worse than that of CNF (0.0508, Table 8.3).

Table 8.6: Retrieval performance (measured by *MAP*) of $P(t|R)$ **guided bag of word expansion** on TREC 4 Ad hoc track, by selecting $n$ query terms for expansion and expanding each of the selected query terms with at most $m$ manual expansion terms. The baseline unexpanded queries produced an MAP of 0.1973. Reported are the best performance of the uniform and Relevance Model weighting runs.

| $\backslash\, n$ <br> $m\backslash$ | 1 | 2 | 3 | 4 | All |
|---|---|---|---|---|---|
| 1 | 0.2101** | 0.2102* | 0.2117** | 0.2113** | 0.2113** |
| 2 | 0.2146*** | 0.2161*** | 0.2200** | 0.2201** | 0.2201** |
| 3 | 0.2160*** | 0.2154*** | 0.2222*** | 0.2226*** | 0.2218** |
| 4 | 0.2204# | 0.2272*** | 0.2288*** | 0.2309** | 0.2309** |
| All | 0.2215# | 0.2290** | 0.2329** | 0.2343** | 0.2384** |

Table 8.6 shows the *best case* bag of word expansion results on the TREC 4 Ad hoc dataset. Consistent with the statAP measure on TREC Legal 2007, CNF queries are much better than bag of word expansion. For example, with full expansion of all query terms, CNF expansion (Table 8.4) gets a MAP of 0.2938, 23% better than 0.2384 of the bag of word expansion with the same expansion terms, significant at $p < 0.0025$ by the randomization test and weakly significant at $p < 0.0676$ by the sign test.

Some results of bag of word retrieval at low selection levels, i.e. selecting one query term to expand, perform better than idf guided CNF expansion. But since the bag of word expansion here uses better expansion terms selected by predicted $\hat{P}(t|R)$, this does not mean that bag of word is sometimes better

than CNF expansion.

The *second* bag of word expansion baseline is the standard Lavrenko Relevance Model itself (Lavrenko and Croft 2001), which uses automatic feedback terms, instead of manual ones, for expansion. Parameters trained on Legal 2006 dataset when applied to Legal 2007 lead to an MAP of 0.0606, statAP of 0.0168, worse than the no expansion baseline. On TREC 4, it gets a MAP of 0.2488, slightly better than 0.2384, the best manual bag of word expansion, but still much worse than CNF (0.2938).

In sum, given the same set of high quality expansion terms, CNF expansion works much better than bag of word expansion.

## 8.6 Summary

We set out with the hypothesis that term mismatch based diagnosis can successfully guide further retrieval intervention to fix 'problem' terms and improve retrieval. In this work, we applied the term mismatch diagnosis to guide interactive query expansion. Simulated interactive query expansion experiments on TREC Ad hoc and Legal track datasets not only confirmed this hypothesis, but also showed that *automatically predicted* $P(t|R)$ probabilities (the complement of term mismatch) can accurately guide expansion to the terms that need expansion most, and lead to better retrieval than when expanding rare terms first. From the user's point of view, it usually isn't necessary to expand every query term. Guided by predicted $\hat{P}(t|R)$, expanding two terms is enough for most queries to achieve close-to-top performance, while guided by idf (rareness), three terms need to be expanded. $\hat{P}(t|R)$ guidance can save user effort by 33%.

In addition to confirming the main hypothesis, experiments also showed that Boolean conjunctive normal form (CNF) expansion outperforms carefully weighted bag of word expansion, given the same set of high quality expansion terms. The unstructured bag of word expansion typically needs balanced expansion of most query terms to achieve a reliable performance. Furthermore, expert CNF expansion can lead to 50-300% gain in retrieval accuracy over baseline short keyword queries.

Although the effect from adding more expansion terms to a query term diminishes, for the query terms that do need expansion, the effects of the expansion terms are typically additive, the more the expansion the better the performance. This is consistent with prior observations on vocabulary mismatch, that even after including more than 15 aliases, the effects of mismatch can still be observed, and further expansion may still help (Furnas et al. 1987).

For bag of word expansion, including more manual expansion terms also helps, but requires a balanced expansion of most query terms, and is not as effective and stable as CNF expansion.

Given the unweighted CNF queries and the term weighting work in the chapters above, it is natural to ask whether one can weight the conjuncts in the CNF queries to further improve retrieval performance. We measured the potential of this approach by using the true term recall probabilities of the expanded conjuncts to weight the conjuncts. However, both because of the effects of expansion and because there are only a small number of conjuncts in a query, i.e. a small number of weights to tune, the potential gain is only around 5%. Thus, it may not be worthwhile to weight the queries which only contain a small number of conjuncts, and CNF expansion is much more effective than weighting.

This work is mostly concerned with automatic diagnosis of problem terms in the query, and presents them to the user for manual expansion. It is still a question whether the diagnosis can help automatic formulation of effective CNF queries. We hope to let the system suggest or select expansion terms, automatically or semi-automatically with minimal user effort. Automatic identification of high quality expansion terms would be useful when the candidate expansion terms may not be of high quality, e.g. expansion terms from result documents, thesaurus or non-expert users. Poor expansion terms in CNF queries are especially harmful, when they over-generalize the query and introduce false positives throughout the rank

list. Tools such as performance prediction methods (e.g. query clarity), may help in such scenarios to detect the adverse effects of the poor expansion terms.

In the future, we also hope to diagnose precision related problems as well as mismatch problems. We can then use the diagnosis to guide a wide variety of techniques such as disambiguation, phrasing or fielded retrieval, as well as term substitution, removal or expansion.

# Chapter 9

# Conclusions and Future Work

This dissertation defines, analyzes, predicts and tries to solve the term mismatch problem in retrieval. Since mismatch is a central problem in retrieval, this dissertation has implications in retrieval theory, as well as practice.

This conclusion chapter is organized as follows. Section 9.1 summarizes the main results and the contributions of this work. Section 9.2 summarizes the fundamental ideas developed in this work and their importance, e.g. how they improve our understanding about retrieval and retrieval models and how they suggest new and effective interventions in retrieval. Section 9.3 shows how future research can more easily develop new and better techniques to solve retrieval and related problems, based on the foundation created in this dissertation.

## 9.1 Main Results

The dissertation results are centered around the term mismatch probability, which is defined as the probability of a term $t$ not appearing in the set $R$ of documents relevant to the query – $P(\bar{t}|R)$.

We show how term mismatch is important for retrieval both theoretically and practically. We show conclusions from data analyses aiming to understand why some query terms have a high mismatch probability. We design effective and efficient ways of predicting term mismatch for a test query without knowledge of the relevant set. We finally show how predicted term mismatch probabilities can be used to improve retrieval in principled ways that would lead to substantial gains for ad hoc full-text retrieval.

### 9.1.1 Term Mismatch and Information Retrieval

Different from prior research, the thesis research defines term mismatch formally and quantitatively, and by doing so, it identifies the central role that term mismatch plays in probabilistic retrieval theory and common retrieval models (Chapter 3).

In theory, the complement of term mismatch, the term recall probability $P(t|R)$, is one of the two class-conditional probabilities that determine the Bayesian optimal term weight – the Robertson-Spärck-Jones (RSJ) weight – which assesses term importance in retrieval. Term recall is also the only component in a retrieval model that is about relevance. The other class-conditional probability $P(t|\bar{R})$, the other component of the RSJ weight, has led to the traditional term weighting based on idf (rareness). Since the notion of relevance is dependent on the particular query under consideration, the tf and idf components of a retrieval model, being independent of the query, have nothing to do with relevance. Thus, term recall is

an important part of the retrieval model; it assesses term importance to a particular information need, and is the only part that produces relevance based term importance.

In practice, this understanding allows us to explain important limitations of the current retrieval models. We show how term mismatch leads to suboptimal retrieval by causing the retrieval model to assign incorrect importance to query terms, which in turn causes false positives (irrelevant documents) which happen to contain the high idf query terms (many times) to appear at higher ranks than the true relevant results that mismatch part of the query. Simply inserting the $P(t|R)$ probabilities back into the term weights leads to 30% to 80% gain in retrieval accuracy. This false positive problem is called *the emphasis problem*, and is one of the most common reasons for the current retrieval models and techniques to fail on long queries, as shown in the large scale failure analysis performed at the 2003 RIA workshop. The RIA workshop identified another problem, the mismatch problem, as a separate problem from the emphasis problem. *The RIA mismatch problem* is defined as one that can only be solved through query expansion. We now know that actually both problems are caused by term mismatch, which, if solved completely, would lead to more than 30-80% retrieval performance gain through term weighting techniques alone and more than 50-300% gain through Conjunctive Normal Form query expansion.

Since term mismatch is a long standing problem in retrieval, many retrieval techniques have been proposed to solve mismatch. This dissertation also summarizes, in Chapter 2, the prior techniques that addressed term mismatch, for example, full text indexing, using inlink anchor texts to enrich hypertext documents, stemming and query expansion. These retrieval techniques which reduce term mismatch tend to be the most reliable techniques with wide adoption, while techniques that increase the chances of term mismatch, e.g. by increasing term precision using more restrictive query term matching, do not always result in reliable performance gains.

### 9.1.2 Analyzing $P(t|R)$

Given the formal definition of term mismatch, the dissertation uses exploratory data analyses to show the prevalence of the term mismatch problem, to analyze the variation of the term mismatch probability, and to identify factors that may cause high mismatch, or factors that may cause the probability of mismatch to vary for the same term across different queries. We focus on analyzing for query terms, however, term $t$ here does not need to be a query term, neither a natural language word. A term is in general a binary function that maps a collection document to 0 or 1.

Exploratory data analyses show that the average query term, in both short or long queries will mismatch (not appear in) 30-40% of the relevant documents for the query. Many query terms suffer from the mismatch problem, and mismatch is quite prevalent. Furthermore, the term mismatch probability varies widely from 0 to 1 across different terms, demanding to be modeled (Chapter 3).

In an effort to understand the underlying mechanisms that cause mismatch, a set of factors have been identified as possible causes of mismatch. These factors include the term not being central to the topic of the query, the concept represented by the term not being necessary for the query, the term having many synonyms that appear in place of the original term, and abstract terms being replaced by more concrete terms in the relevant documents. Because simple collection statistics do not correlate well with term mismatch, and no effective features have been identified by prior research that predicted mismatch, the possible causes of mismatch identified in Chapter 3 form the basis for the design of effective features that can predict the term mismatch probability.

By definition, the term mismatch probability is dependent on the term and the query that the term appears in. To understand how mismatch varies for the same term across different queries, a set of data analyses looked at the repeating terms, the query terms that appear in more than one query in the datasets, and how their term mismatch probabilities vary (Chapter 4). Analyses show that in practice, the term

mismatch probability does vary in a query dependent way for the same term. However, at the same time, for many repeating occurrences, the term mismatch probability does not change very much from the term mismatch probabilities of the historic occurrences. Thus, even though query dependent features are necessary for effective $P(t|R)$ prediction, it may be effective to use information from historic occurrences of a term to predict the $P(t|R)$ probability for a new occurrence.

Data analyses also show that factors such as differences in word sense, word use or query semantics can cause large variations of the term mismatch probability for the same term in different queries, but frequently cross-query variations may exist even without a change in word sense or use or query semantics. We attribute these cases of variation to the term's level of association with the query, a common cause of $P(t|R)$ variation.

### 9.1.3 Predicting Term Mismatch and its Variation

Given the data analyses, the dissertation focuses on designing methods to effectively and efficiently predict term mismatch for query terms. Numeric features were designed to model the factors that cause mismatch or cause its query dependent variation, were found to correlate well with term mismatch or its variation, and were used to learn prediction models to predict term mismatch for test queries with no relevance judgments.

The thesis research designed an effective 2-pass term mismatch prediction method based on the possible causes of mismatch identified during data analyses (Chapter 5). This 2-pass method only uses the query and the document collection to identify synonyms of query terms and to predict term mismatch. No external resources are used. Synonyms are identified in a query dependent way using local LSI, i.e. for each query, using an initial keyword retrieval followed by latent semantic indexing over the top ranked documents to create a latent semantic space to measure term similarity. Numeric features are then designed based on the automatically identified synonyms to model the factors that cause term mismatch. These automatic features were found to correlate well with the prediction target, term mismatch, and were used to predict term mismatch on test queries using a regression model learnt from a training set with known relevance. The new features performed much better than only using idf to predict $P(t|R)$ in prior research.

Through these experiments, this dissertation research has created a general framework in which it is easy to design and test new features that correspond to new hypotheses about causes of mismatch or low recall.

Because of the sequential steps of initial keyword retrieval, extracting information from the top ranked documents, LSI computation and synonym extraction, the 2-pass prediction approach can take more than 10 seconds per query on a collection containing about 25 million Web pages. This is not efficient enough for low-latency search environments. We have designed an efficient 1-pass term mismatch prediction method that does not require an initial retrieval nor LSI, but is based on the observation that the term mismatch probability does not vary much for the same term in different queries (Chapter 7). We use the known $P(t|R)$ probabilities from historic occurrences of the term in training queries that are similar to the test query to predict for the new occurrence of the term in the test query. This efficient 1-pass method achieved a prediction accuracy close to that of the more expensive 2-pass method, but at the same time being efficient enough for real-time response scenarios. The 1-pass prediction method requires the training query terms to cover a large enough portion of the test query terms to be effective. When the training set is small, to increase the test term coverage of the 1-pass prediction method, a bootstrapping method is designed. During offline computation, it uses the small training set (with relevance judgments) to train a 2-pass prediction model, and then uses the 2-pass model to automatically predict $P(t|R)$ for a large set of queries from a query log. During online computation, the large number of query terms from the query

log with automatic predictions can be used to train and predict $P(t|R)$ using the efficient 1-pass method, largely increasing test term coverage.

### 9.1.4 Solving Term Mismatch Using Mismatch Predictions

Based on the new understanding of the mechanism of how term mismatch affects retrieval performance, the dissertation applies the term mismatch predictions in different ways to solve mismatch and improve retrieval. Term weighting and term expansion are two such retrieval interventions directly motivated by the theory, and are investigated in detail.

Since the probability $P(t|R)$ appears in probabilistic retrieval models as part of the term weight, the $P(t|R)$ predictions can be directly used as term weights in these probabilistic retrieval models (Chapter 6). Experiments on 6 different test sets from TREC Ad-hoc and Web search tracks show that for long queries, reweighting query terms with their predicted $P(t|R)$ outperforms the baseline idf-weighted keyword retrieval models in both retrieval recall and precision. The supervised $P(t|R)$ predictions also outperform the unsupervised relevance model weights. The efficient 1-pass $P(t|R)$ prediction method achieves a performance close to that of the slower 2-pass prediction method. Sometimes, with enough coverage over test terms, the 1-pass method can achieve a higher retrieval accuracy than the 2-pass method.

Term expansion (Chapter 8) is another retrieval intervention to solve mismatch. It aims to solve term mismatch by expanding each query term with its synonyms, so that the overall recall of that synonym group – $P(t$ OR synonyms of $t|R)$ – would improve over $P(t|R)$ – the term recall of the original term alone. Expanding query terms individually results in a Boolean Conjunctive Normal Form (CNF) query. Experiments on TREC Ad-hoc and Legal tracks show that with high quality manual expansions, CNF style expansion outperforms keyword baseline by 50-300%, as well as outperforming bag of word expansion with the same set of high quality expansion terms. The best CNF expansion performance is achieved when fully expanding all of the query terms with all of the manually identified synonyms. To save user expansion effort, a diagnostic intervention approach is used, where $P(t|R)$ predictions are used to identify the query terms that are likely to mismatch and need expansion most. Expanding only the two query terms with the highest likelihood of mismatch achieves most of the benefit of CNF expansion, saving 33% of user effort compared to the competing method of expanding rare (high idf) terms first.

## 9.2 Significance

This work improves our understanding and makes progress toward solving two central and long standing problems in retrieval. We summarize the significance of the work by explaining how these new understandings allow us to explore new and principled interventions to make progress on these two significant yet difficult problems, and to improve retrieval models and techniques.

### 9.2.1 The $P(t|R)$ Estimation Problem and the Term Mismatch Problem

$P(t|R)$ is known as an integral part of probabilistic retrieval models, the only part that is in effect about relevance. Even though it was known that accurate estimates of $P(t|R)$ can lead to significant improvements in retrieval (Robertson and Spärck Jones 1976), prior research found it difficult to estimate $P(t|R)$ and ended up using idf as the only feature for prediction (Greiff 1998; Metzler 2008). Term mismatch (Furnas et al. 1987) is another well known problem in retrieval. Even though different techniques have been proposed to solve the vocabulary mismatch between the query and the relevant documents, there was no clear definition and understanding of term mismatch, neither how term mismatch affects retrieval performance and how different techniques can solve mismatch. This dissertation improves our understanding

on these two problems, advances the techniques to solve them, and identifies areas of future research that would not have been possible without the new understandings.

This dissertation formally connects these two problems, fundamentally changes how the research field understands these two problems, and has made the two problems much easier to approach. This research shows that the term mismatch probability is directly related to $P(t|R)$ – it is the complement of $P(t|R)$. This means term mismatch is an integral part of common retrieval models, but is often ignored due to estimation difficulty. This connection allows us to understand how term mismatch affects retrieval performance through the emphasis and mismatch problems identified in the RIA workshop (Harman and Buckley 2009), and allows us to explain many of the behaviors of retrieval models and techniques. This connection has also led to effective and principled ways to predict $P(t|R)$, solve mismatch and improve retrieval.

First, this new understanding allows us to understand how different retrieval techniques interact with the retrieval models. Previously, the retrieval models are similar to each other, being based on simple collection statistics, and some retrieval techniques applied on top of the retrieval models can reliably improve over the retrieval models, while others cannot. These interactions between the different retrieval techniques and the retrieval models have just become part of the standard knowledge of the field of information retrieval, with no clear understanding of why these techniques behave so.

This new understanding of retrieval models and term mismatch can help us explain why the techniques behave so. For example, when combining 2-grams with 1-grams in a weighted query, 2-grams usually should be given a much lower weight than the 1-grams. This is perhaps because 2-grams mismatch more, and have a lower $P(t|R)$ probability, thus, lower optimal weight. Furthermore, such phrases or n-grams only improve retrieval when carefully combined with the 1-grams, and even when carefully used, they still tend to only increase top precision while decreasing retrieval accuracy at lower ranks. This is perhaps because phrases and n-grams increase the precision of the matching, leading to an increase in precision at the top ranks when there are enough relevant documents in the collection that match the phrase query. But at the same time, they cause more mismatch, lowering overall retrieval performance, especially at the lower ranks. Another example, other techniques that aim to make queries more precise, e.g. fielded retrieval based on syntactic or semantic annotations, word sense disambiguation or personalization, are usually unstable and do not show consistent improvement over the keyword retrieval baseline. These high precision retrieval techniques typically cannot improve all queries. This is perhaps because many of the queries suffer from the mismatch problem instead of the precision problem, thus increasing the strictness in matching can only make the situation worse. Indeed, techniques that try to reduce the field level mismatch were shown to outperform simply using the original query structure for retrieval (Zhao and Callan 2009).

Second, this new understanding enables us to investigate theoretically motivated techniques that solve mismatch and improve retrieval. For example, this has led to the 2-pass supervised $P(t|R)$ prediction method which is a general and effective method to predict $P(t|R)$, while previous research either used unsupervised methods that are less effective (e.g. Rocchio style feedback) or relied on external resources for term weight prediction (Lease et al. 2009). Our 1-pass log based $P(t|R)$ prediction method uses query log data in an instance based learning framework. Furthermore, for retrieval performance, $P(t|R)$ based term weighting is one way to address mismatch that does not involve expansion terms. Conjunctive Normal Form expansion is another way that expands and improves the term recall of individual query terms, instead of expanding the query as a whole. Both term weighting and expansion techniques can lead to consistent and significant gains over the standard retrieval models. Unlike techniques that only enhance top precision but can hurt lower level precision, the gains here are typically observed at all recall levels.

### 9.2.2 A General Definition of Mismatch and a Wide Range of Analyses

This dissertation defines term mismatch in a general way, so that the methodology developed in this dissertation can be applied to all standard datasets used in retrieval evaluation. This has an immediate relevance to the current information retrieval research, because a wide range of retrieval datasets have been made available through evaluation forums like TREC, CLEF, NTCIR and FIRE. These standard retrieval collections with queries and relevance judgments can be used in a straightforward way to estimate term mismatch and quantitatively analyze mismatch, without any dependency on external resources. These data analyses allow researchers to understand whether mismatch is an important problem for a particular retrieval task given its corresponding dataset.

In this work, the term mismatch related data analyses allow us to infer the possible causes of mismatch, to design numeric features that model the causes, and to predict mismatch. Feature quality can be easily evaluated using its correlation with the prediction target – the term mismatch probability, or using prediction and retrieval experiments. This whole framework makes it easy to test new hypotheses about causes of mismatch, and to design effective features and methods to predict term mismatch and improve retrieval.

### 9.2.3 Diagnostic Interventions

Retrieval techniques that try to improve a query are often restricted to the use of query expansion or query suggestion. These techniques are typically applied to the whole query or to all the query terms without identifying which part of the query or which query terms are causing trouble.

Diagnostic intervention adds a useful dimension to the current query improvement techniques. Traditional query expansion techniques typically produce long, complex and unstable expansion queries. The diagnosis based interventions provide a way for the user to effectively interact with the retrieval system, and typically lead to simple, effective and robust expansion queries. Traditional query suggestion techniques aim to fix the problems of the original query by directly suggesting several alternative queries for the user to choose from, and are a low cost way to elicit more information from the user. The diagnosis based interventions have a different goal, aiming to find out where the problem might be for a given query before trying to fix and improve the query. Thus, the interventions tend to be more effective with the correct diagnosis.

There have been query performance prediction methods in the literature, but seldom followup work that can use these predictions to guide further interventions. These performance prediction measures do correlate well with overall retrieval performance, however, they do not identify clearly what kind of problem a particular query is suffering from, and the diagnosis is perhaps not detailed enough to be used in meaningful ways. The diagnosis in this work targets term mismatch problems, and can be effectively used to guide mismatch-related interventions to improve retrieval reliably.

### 9.2.4 Boolean Conjunctive Normal Form Structured Queries

We revisit the Boolean conjunctive normal form (CNF) queries, which were shown to be less effective than the unstructured keyword queries in the 1990s. Since the 1990s, the research community has mostly steered away from the Boolean structured queries and has focused on the relatively unstructured bag of word queries for query expansion and formulation. However, search experts from the industry (e.g. lawyers and librarians) still prefer Boolean structured queries. A disconnect exists between the industrial practice and the research efforts.

This dissertation research tries to close this gap by demonstrating a new way of creating Boolean queries that can be much more effective than bag of word querying. First, we observe that the prior exper-

iments that showed favorable results for the keyword queries typically compared against unranked Boolean retrieval. Using Boolean structured queries in a ranked retrieval setup can change the picture. Second, bag of word expansion is favored by the research community and in evaluation forums like TREC, because of its average case performance, where adding many expansion terms into a bag of word query usually shows an average improvement over the unexpanded keyword baseline. However, the improvements on individual queries are typically unstable. Thus, bag of word expansion is not widely adopted in industry. Our results show that by being selective and using high quality expansion terms, the Boolean CNF structured queries can significantly outperform unstructured bag of word queries, leading to more robust improvements than the unstructured queries. This result points at the direction that expert searchers take, where research is scarce and much in need.

## 9.3 Future Work

Understanding term mismatch and its role in retrieval modeling provides a necessary and unique tool for future interventions to solve mismatch and improve retrieval. Future research can apply the new understandings and ideas developed in this work in new ways that would not have been possible before.

### 9.3.1 Modeling Mismatch

The dissertation research provided an initial set of data analyses into the possible causes of mismatch. Currently, only a small set of causes have been identified, and only some fairly coarse features have been designed and used to model the causes and predict mismatch. For example, the synonyms/searchonyms automatically identified by local Latent Semantic Indexing (LSI) are fairly noisy. The current abstractness feature only provides a binary guess about whether a term is likely to be abstract. There are no levels of abstractness, nor what more specific terms may appear in place of that abstract term in relevant documents.

We know from this dissertation research that more accurate predictions of the term mismatch probability can lead to even larger gains in retrieval. To achieve more accurate predictions, it is key to accurately model a comprehensive set of factors that could cause mismatch. Further data analyses based on the framework designed in this work can identify a more complete set of causes of mismatch, as well as how much each of them contribute to term mismatch overall.

Future research can also build upon this research by for example analyzing each query term and mismatched relevant document pair, or performing failure analyses of the current mismatch prediction techniques to provide more accurate synonyms and more effective prediction features.

Our research also reveals a theoretical inconsistency in the modeling of term mismatch (or equivalently, term relevance) in the language modeling framework. Currently, term relevance $P(t|R)$ is found to be best modeled as a Bernoulli distribution disregarding the term occurrence frequencies in the relevant documents. However, at the same time, document models are best captured in multinomial or multiple-Bernoulli (Model B from (Metzler et al. 2004)) distributions, which do take into account term frequency information from the returned documents. These inconsistent assumptions about the underlying term distributions demand better interpretation of the current models or more theoretically consistent modeling of term distributions. Accumulations of such theoretical inconsistencies usually lead to improvements in the theory, and sometimes even paradigm shifts.

Overall, all these analyses can lead to better features, improved term mismatch prediction and better understanding and modeling of term relevance.

### 9.3.2 Solving Mismatch with Conjunctive Normal Form Expansion

Query expansion can solve mismatch from the root, by improving the term matching between the individual query terms and the relevant documents. Search request formulation techniques like query expansion has long been recognized as the component of the retrieval system that has the largest potential variance (Rocchio 1971). The term mismatch theory suggests the use of conjunctive normal form (CNF) style query expansion, which is less well studied, but shown in this dissertation to be able to outperform the traditional bag of word style expansion when given the same set of high quality expansion terms. Based on this result, future research can explore automatic and manual CNF expansions to suit the needs of different use cases. Possible directions include identifying high quality synonyms or related terms based on the local context of the query, or designing retrieval models that could handle noisy expansion terms.

Current CNF expansion techniques range from purely manual to purely automatic. However, the middle ground is less well developed, where interactive interfaces facilitate users in formulating effective CNF queries. New tools are needed to help users manually identify high quality expansion terms and improve the effectiveness of interactive search, to propose and present candidate expansion terms from feedback documents or other sources to the user, or to analyze the effects of these expansion terms and effectively reveal the impact of the individual expansion terms to the user.

Currently, query expansion based on static thesauri is not as effective as pseudo relevance feedback based on the query context. This is likely because of the query dependent nature of term mismatch and synonyms/searchonyms of query terms. Future research may apply our query dependent mismatch prediction framework to dynamically utilize a comprehensive set of thesauri and related resources, and to fit them to the local contexts of the queries.

Currently, automatically identified synonyms are noisy, often containing false synonyms. When expansion terms can be noisy, novel weighting mechanisms for incorporating these synonyms during retrieval can help reduce the adverse effect of the false synonyms while still maintaining the benefits of the good expansion terms. In order to achieve the better weighting for synonyms, novel retrieval models that can model the effects of synonym expansion are needed. Synonym weighting in Boolean structured queries is a new direction that prior research did not focus on. Traditionally, term weighting is applied to weight query terms or expansion terms in a weighted conjunction (AND). Future research can build on top of our research on ranked retrieval using CNF queries, and investigate the use of disjunctive (OR-based) term weighting, to combine the expansion terms of a particular query term in an effective and robust way.

An alternative to the Boolean CNF expansion is to include whole paraphrases. For manual query formulation, using whole paraphrases may not be as compact as CNF queries. However, for automatic query expansion, disjunctions of whole paraphrases or whole query reformulations may be more robust in achieving better retrieval performance than per term expansion in the CNF style. It is still unclear to the IR research community which approach might work better under what circumstances, or whether it is possible to convert a set of paraphrases into a general and effective CNF query.

### 9.3.3 Retrieval and Natural Language Understanding

It is perhaps well accepted that deeper natural language understanding is needed for perfect retrieval. Thus, to achieve better retrieval performance beyond this dissertation and the current state of the art, it might be necessary to have the retrieval system understand natural language in deeper ways than just the simple tf or idf statistics. However, for years, it has been shown that complex natural language understanding techniques do not show a stable improvement over the simple tf.idf retrieval models. These two pieces of knowledge are seemingly contradictory, because it was never clear what part of a retrieval model would require any deep understanding of the natural language texts.

Prior research largely overlooked the term mismatch problem, and used natural language processing (NLP) to improve term precision, e.g. by requiring certain words to be of a certain word sense, from a certain constituent or being modified by certain other query terms. These techniques could not result in stable performance gains, because such restrictions although they do increase term precision, can reduce term recall significantly, causing mismatch and decreasing retrieval accuracy. On the other hand, techniques aimed to reduce mismatch, such as stemming, term reweighting for long queries, term abstractness measures, Boolean CNF expansion or translation based retrieval models, are found to be helpful for retrieval consistently.

We show that to effectively solve term mismatch, a deeper understanding of the query and the document texts seems necessary. We have successfully applied dependency parsing and latent semantic analysis to generate effective features for term mismatch prediction. Our framework for designing and evaluating mismatch prediction features has made it easy to discover better ways to apply NLP techniques and to design NLP tools that specifically target term mismatch, an important problem in retrieval. In particular, future research may explore and apply models of natural language variation to capture the large textual variations of the texts that convey the same meaning. Examples of such techniques include models used for predicting translational equivalence, paraphrasing or textual entailment. A more radical approach is to use semantic modeling of natural language texts to normalize the textual variation into a standard semantic representation which could then be used to reduce mismatch. These NLP techniques will likely lead to more accurate and flexible relevance matching, and get us closer to perfect retrieval.

### 9.3.4 Diagnosis for Effective Retrieval Interventions

A key idea in this dissertation is diagnostic intervention. The dissertation research diagnoses the mismatch problem at the term level and guides further expansion interventions to the terms that are likely to mismatch relevant documents.

Further applications of the diagnosis idea can identify specific types of mismatch problems or even identify different types of problems.

This research shows that mismatch can be caused by a concept not being central to the topic of the query or by synonyms or hyponyms replacing the original query term. These different causes of mismatch specify different types of mismatch problems. The dissertation research only uses the causes to generate features to predict mismatch. Future research may focus on identifying the specific types of mismatch problems suffered by a particular query. The more detailed diagnosis allow different interventions to be designed and applied to solve these different types of mismatch cases, further improving prediction and retrieval accuracy.

The dissertation focuses on the term mismatch problem, but a query can suffer from a number of different problems e.g. emphasis, mismatch, or precision problems. Being able to diagnose different kinds of problems at the different levels of the retrieval process can guide the application of a wide variety of retrieval techniques that are aimed to solve these different problems, potentially leading to more effective uses of the current or future retrieval techniques. The diagnostic interventions allow different retrieval techniques to be applied selectively on a session-by-session, query-by-query or term-by-term basis, according to the needs of the sessions, queries or terms, instead of uniformly in all cases.

### 9.3.5 A General View of Retrieval Modeling as Meta-Classification

Our framework of mismatch and term weight prediction demands a more general view of the retrieval modeling problem as a transfer learning problem (Do and Ng 2005). Prior research typically took a

restricted view of retrieval as a document classification problem: Given a query, classify collection documents according to their relevance to the query.

We view the retrieval model as a meta-classifier, which takes a query as input and produces a document classifier as output. The classifier produced by the retrieval model is then used to rank the collection documents. Based on this more general view, learning such a retrieval model from a set of training queries becomes a transfer learning task, where training queries provide related document classification tasks, so that the knowledge about relevance ranking learnt on these training tasks is transferred to produce a new document classifier for the test query. For example, the more efficient query log based 1-pass $P(t|R)$ prediction method can be seen as a direct transfer of knowledge about $P(t|R)$ from query terms in training queries to improve retrieval for the same term in a test query.

Transfer learning is a new formalism for learning effective retrieval models, and has several advantages over existing formalisms. Compared to learning to rank which only employs one global classifier for all queries, one advantage of the transfer learning formalism is its flexibility to generate a completely new classifier for each test query. This flexibility makes it natural to apply the formalism in adaptive scenarios such as relevance feedback, while other single-classifier formalisms (such as learning to rank) would at least need to re-train the entire model when given an additional feedback document for a test query. Compared to learning to rank, especially pairwise or list-wise learning to rank methods, which requires a large number of training samples for each query, transfer learning has the advantage of showing improvements with only a small number of training samples per task (Do and Ng 2005). The traditional formalism of each query as a separate classification task does make it easy to use in relevance feedback tasks. But compared to this traditional view, the transfer learning formalism has the benefit of allowing the use of relevance judgments from unrelated training queries to be used to improve the performance of an unseen test query. Overall, the transfer learning formalism is more flexible, and shares many of the advantages of the existing formalisms.

Given this general view of retrieval modeling as transfer learning, future studies can carry out principled investigations of the retrieval modeling problem under this new formalism, and apply advanced transfer learning techniques in retrieval. All of these could improve our understandings about retrieval modeling as well as state-of-the-art retrieval accuracy.

# Bibliography


James Allan, Lisa Ballesteros, James P. Callan, W. Bruce Croft, and Zhihong Lu. Recent experiments with INQUERY. In *Proceedings of the 4th Text REtrieval Conference - TREC '95*. National Institute of Standard Technology, 1996. 2.3.1

Aris Anagnostopoulos, Andrei Z. Broder, and Kunal Punera. Effective and efficient classification on a search-engine model. In *Proceedings of the 15th International Conference on Information and Knowledge Management - CIKM '06*, pages 208–217. ACM Press, 2006. 5.4, 2

Javed A. Aslam and Virgil Pavlu. A practical sampling strategy for efficient retrieval evaluation. *Report*, 2007. URL `http://www.ccs.neu.edu/home/jaa/tmp/statAP.pdf`. 8.4.6

Jason R. Baron, David D. Lewis, and Douglas W. Oard. TREC 2006 Legal track overview. In *Proceedings of the 15th Text Retrieval Conference - TREC '06*. National Institute of Standard Technology, 2007. 2.6.3.2, 2.6.3.2, 3.6.4, 8.2.2, 8.4.3.1

Michael Bendersky and W. Bruce Croft. Discovering key concepts in verbose queries. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '08*, pages 491–498, New York, NY, USA, 2008. ACM Press. 2.4, 2.4.4, 2.5, 2.6.4, 2.7.1.2, 2.7.3

Michael Bendersky and W. Bruce Croft. Analysis of long queries in a large scale search log. In *Proceedings of the 2009 Workshop on Web Search Click Data - WSCD '09*, pages 8–14, New York, NY, USA, 2009. ACM. 3.4

Michael Bendersky, Donald Metzler, and W. Bruce Croft. Learning concept importance using a weighted dependence model. In *Proceedings of the third ACM International Conference on Web Search and Data Mining - WSDM '10*, pages 31–40, New York, NY, USA, 2010. ACM. 2.4.3

Adam Berger and John Lafferty. Information retrieval as statistical translation. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '99, pages 222–229, New York, NY, USA, 1999. ACM. 2.6.6.2

J. Bhogal, A. Macfarlane, and P. Smith. A review of ontology based query expansion. *Information Processing & Management*, 43:866–886, 2007. 2.6.2

Matthew W. Bilotti, Paul Ogilvie, Jamie Callan, and Eric Nyberg. Structured retrieval for question answering. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '07*, pages 351–358, New York, NY, USA, 2007. ACM Press. 2.7.3

David C. Blair. STAIRS redux: Thoughts on the STAIRS evaluation, ten years after. *Journal of the American Society for Information Science*, 47(1):4–22, 2004. 8.2.2

David C. Blair and M. E. Maron. An evaluation of retrieval effectiveness for a full-text document-retrieval

system. *Commun. ACM*, 28(3):289–299, March 1985. 3.6.4

Thorsten Brants. Natural language processing in information retrieval. In *Proceedings of the 14th Meeting of Computational Linguistics in the Netherlands*, pages 1–13, 2004. 2.7.3

Chris Buckley and Gerard Salton. Optimization of relevance feedback weights. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '95, pages 351–357, New York, NY, USA, 1995. ACM. 6.2.3.2

James P. Callan, W. Bruce Croft, and Stephen M. Harding. The INQUERY retrieval system. In *Proceedings of the Third International Conference on Database and Expert Systems Applications*, pages 78–83. Springer-Verlag, 1992. 2.3.1

Guihong Cao, Stephen Robertson, and Jian-Yun Nie. Selecting query term alterations for Web search by exploiting query contexts. In *Proceedings of 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - ACL-08: HLT*, pages 148–155, 2008. 2.6.2, 2.6.4, 8.2.2

Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*, ICML '07, pages 129–136, New York, NY, USA, 2007. ACM. 2.4.2

Sharon A Caraballo and Eugene Charniak. Determining the specificity of nouns from text. In *Proceedings of the 1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora - EMNLP/VLC '99*, 1999. 5.2.2.4

David Carmel, Elad Yom-Tov, Adam Darlow, and Dan Pelleg. What makes a query difficult? In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '06*, pages 390–397, New York, NY, USA, 2006. ACM Press. 2.8

Soumen Chakrabarti, Byron Dom, Prabhakar Raghavan, Sridhar Rajagopalan, D. Gibson, and J. Keinberg. Automatic resource list compilation by analyzing hyperlink structure and associated text. In *Proceedings of the Seventh International World Wide Web Conference - WWW '98*, pages 65–74, 1998. 2.6.1

Charles L. A. Clarke, Gordon V. Cormack, and Forbes J. Burkowski. Shortest substring ranking (MultiText experiments for TREC-4). In *Proceedings of the 4th Text REtrieval Conference - TREC '95*. National Institute of Standard Technology, 1996. 8.2.2, 8.4.3.2

Kevyn Collins-Thompson. *Robust model estimation methods for information retrieval*. PhD thesis, Language Technologies Institute, Carnegie Mellon University, 2008. 2.6.3

William S. Cooper, Fredric C. Gey, and Daniel P. Dabney. Probabilistic retrieval based on staged logistic regression. In *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '92*, pages 198–210, New York, New York, USA, 1992. ACM Press. 2.4.1

Gordon V. Cormack, Charles L. A. Clarke, Christopher R. Palmer, and Samuel S. L. To. Passage-based refinement (MultiText experiments for TREC-6). In *Proceedings of the 6th Text REtrieval Conference - TREC '97*. National Institute of Standard Technology, 1998. 2.6.3.2

Daniel W. Crabtree, Peter Andreae, and Xiaoying Gao. Exploiting underrepresented query aspects for automatic query expansion. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - SIGKDD '07*, pages 191–200. ACM Press, 2007. 2.7.1.2

W. Bruce Croft. An approach to natural language for document retrieval. In *Proceedings of the 10th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*

- *SIGIR '87*, pages 26–32, New York, NY, USA, 1987. ACM. 2.7.3

W. Bruce Croft and David J. Harper. Using probabilistic models of document retrieval without relevance information. *Journal of Documentation*, 35(4):285–295, December 1979. 1, 2.3.1, 2.3.1, 2.6.3

Steve Cronen-Townsend, Yun Zhou, and W. Bruce Croft. Predicting query performance. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '02*, pages 299–306, New York, New York, USA, 2002. ACM Press. 2.8, 5.2.2.6, 7.2.2.2

Van Dang and W. Bruce Croft. Query reformulation using anchor text. In *Proceedings of the 3rd ACM International Conference on Web Search and Data Mining - WSDM '10*, pages 41–50, New York, NY, USA, 2010. ACM Press. 2.6.2, 2.6.4, 2.7.1.2, 8.2.2

Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6): 391–407, 1990. 2.6.6.1, 8.2.1, 8.4.4

Cuong B. Do and Andrew Y. Ng. Transfer learning for text classification. In *Neural Information Processing Systems Foundation*, NIPS '05, 2005. 1.6, 2.5, 5.4, 9.3.5

Henry A. Feild, James Allan, and Rosie Jones. Predicting searcher frustration. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '10, pages 34–41, New York, NY, USA, 2010. ACM. 1

Norbert Fuhr and Chris Buckley. A probabilistic learning approach for document indexing. *ACM Transactions on Information Systems*, 9(3):223–248, July 1991. 2.4.1

George W. Furnas, Thomas K. Landauer, Louis M. Gomez, and Susan T. Dumais. The vocabulary problem in human-system communication. *Communications of ACM*, 30:964–971, November 1987. 1, 2.6.6.1, 3.1, 8.2.1, 8.6, 9.2.1

J. Gao, G. Cao, H. He, M. Zhang, J. Nie, S. Walker, and S. E. Robertson. TREC-10 web track experiments at MSRA. In *Proceedings of the Tenth Text REtrieval Conference - TREC '01*, pages 384–392. National Institute of Standard Technology, 2002. 2.2.5

Jianfeng Gao, Xiaodong He, and Jian-Yun Nie. Clickthrough-based translation models for web search: from word models to phrase models. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, CIKM '10, pages 1139–1148, New York, NY, USA, 2010. ACM. 2.6.6.2

Fredric C. Gey. Inferring probability of relevance using the method of logistic regression. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '94*, pages 222–231, New York, NY, USA, 1994. Springer-Verlag New York, Inc. 2.4.1

Gary Goertz and Harvey Starr, editors. *Necessary Conditions: Theory, Methodology, and Applications*. Rowman & Littlefield, Lanham, Md, 2002. 2.2.3

Warren R. Greiff. A theory of term weighting based on exploratory data analysis. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '98*, pages 11–19, New York, New York, USA, 1998. ACM Press. 1, 1.4, 2.2.2, 2.3.2, 3.4, 3.4.3, 3.6, 5.2.2.5, 5.3.2, 6.2.4.5, 8.1, 8.2.1, 8.3.2, 8.5.2, 9.2.1

Harsha Gurulingappa, Bernd Müller, Roman Klinger, Heinz-Theodor Mevissen, Martin Hofmann-Apitius, Christoph M. Friedrich, and Juliane Fluck. Prior art search in chemistry patents based on semantic

concepts and co-citation analysis. In *Proceedings of the 19th Text REtrieval Conference - TREC '10*, 2010. 2.6.2

Donna Harman. Towards interactive query expansion. In *Proceedings of the 11th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '88, pages 321–331, New York, NY, USA, 1988. ACM. 8.2.3

Donna Harman. Overview of the third text REtrieval conference (trec-3). In *Proceedings of the 3rd Text REtrieval Conference - TREC '94*. National Institute of Standard Technology, 1995. 8.4.3.2

Donna Harman. Overview of the fourth text REtrieval conference (trec-4). In *Proceedings of the 4th Text REtrieval Conference - TREC '95*. National Institute of Standard Technology, 1996. 8.4.3.2

Donna Harman and Chris Buckley. Overview of the Reliable Information Access workshop. *Information Retrieval*, 12(6):615–641, Dec 2009. (document), 2.8, 3.7, 3.6.3, 3.6.3, 6.2.4.3, 9.2.1

Stephen Harter. *Online Information Retrieval: Concepts, Principles, and Techniques*. Academic Press, San Diego, California, 1986. 2.6.3.2, 2.7.1.1, 8.2.2

Marti A. Hearst. Improving full-text precision on short queries using simple constraints. In *Proceedings of the Symposium on Document Analysis and Information Retrieval*, pages 217–228, 1996. 2.6.3.2, 2.7.1.1, 8.2.2, 8.4.5

Randy Burke Hensley and Elizabeth Hanson. Question analysis for autobiography. In *Designs for Active Learning: A Sourcebook of Classroom Strategies for Information Education*, chapter 13, pages 55–58. Association of College and Research Libraries, 1998. 2.7.1.1, 8.2.2

William Hersh and Ellen Voorhees. TREC Genomics special issue overview. *Information Retrieval*, 12 (1):1–15, 2009. 2.7

Dustin Hillard and Chris Leggetter. Clicked phrase document expansion for sponsored search ad retrieval. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval - SIGIR '10*, pages 799–800, New York, NY, USA, 2010. ACM. 2.6.1

David A. Hull. *Information retrieval using statistical classification*. PhD thesis, Department of Statistics, Stanford University, 1994. 5.2.2

Rosie Jones, Benjamin Rey, Omid Madani, and Wiley Greiner. Generating query substitutions. In *Proceedings of the 15th International Conference on World Wide Web - WWW '06*, pages 387–396, New York, New York, USA, 2006. ACM Press. 2.6.3.2, 2.6.4, 8.2.2

Charles Kemp and Kotagiri Ramamohanarao. Long-term learning for Web search engines. In *Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery - PKDD '02*, pages 263–274. Springer, 2002. 2.6.1

April Kontostathis and William M. Pottenger. Detecting patterns in the LSI term-term matrix. In *Proceedings of IEEE ICDM 2002 Workshop on Foundations of Data Mining and Knowledge Discovery (FDM)*, 2002. 5.2.2

Robert Krovetz. Viewing morphology as an inference process. In *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '93*, pages 191–202, New York, NY, USA, 1993. ACM. 2.6.2

Anagha Kulkarni and Jamie Callan. Document allocation policies for selective searching of distributed indexes. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, CIKM '10, pages 449–458, New York, NY, USA, 2010. ACM. 5.3.3

Giridhar Kumaran and Vitor R. Carvalho. Reducing long queries using query quality predictors. In

*Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '09*, pages 564–571, New York, New York, USA, 2009. ACM Press. 2.4, 2.6.4

John Lamping and Steven Baker. Determining query term synonyms within query context. United States Patent No. 7,636,714, March 2005. 2.6.3.2, 8.2.2

Frederick Wilfrid Lancaster. *Information Retrieval Systems: Characteristics, Testing and Evaluation*. Wiley, New York, New York, USA, 1968. 2.6.3.2, 2.7.1.1, 8.2.2

Victor Lavrenko and W. Bruce Croft. Relevance based language models. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '01*, pages 120–127, New York, New York, USA, 2001. ACM Press. 2.1, 2.2.3, 2.2.6, 2.2.6.1, 2.2.6.1, 2.2.6.2, 2.6.3, 4.3, 6.1.2, 6.1.2, 6.1.2.2, 6.1.2.2, 6.2.4.6, 8.2.2, 8.5.4, 8.5.4

Reed C. Lawlor. Information technology and the law. *Advances in Computers*, 3:299–346, 1962. 2.7.2, 5.2.2.2

Matthew Lease. Incorporating relevance and psuedo-relevance feedback in the Markov Random Field model. In *Proceedings of the 18th Text REtrieval Conference - TREC '09*. National Institute of Standard Technology, 2010. 2.6.3.1

Matthew Lease, James Allan, and W. Bruce Croft. Regression Rank: Learning to meet the opportunity of descriptive queries. In *Proceedings of the 31st European Conference on Information Retrieval - ECIR '09*, pages 90–101, 2009. 2.4.3, 2.7.3, 5.2.4.1, 6.2.3, 7.2.2.1, 9.2.1

Whay C. Lee and Edward A. Fox. Experimental comparison of schemes for interpreting boolean queries. *Technical Report TR-88-27, Computer Science, Virginia Polytechnic Institute and State University*, 1988. 2.6.5

Jimmy Lin and Mark D. Smucker. How do users find things with pubmed?: towards automatic utility evaluation with user simulations. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pages 19–26, New York, NY, USA, 2008. ACM. 8.2.3

Tie-Yan Liu, Jun Xu, Tao Qin, Wenying Xiong, and Hang Li. LETOR: Benchmark dataset for research on learning to rank for information retrieval. In *Learning to Rank for Information Retrieval - LR4IR 2007, in conjunction with SIGIR 2007*, 2007. 2.3.1, 2.4.2, 2.4.3

David E. Losada. Language modeling for sentence retrieval: A comparison between multiple-Bernoulli models and multinomial models. In *Information Retrieval and Theory Workshop*, Glasgow, UK, July 2005. 6.1.2.1

Yue Lu, Hui Fang, and Chengxiang Zhai. An empirical study of gene synonym query expansion in biomedical information retrieval. *Information Retrieval*, 12(1):51–68, 2009. 3, 5.2.2.3

Yuanhua Lv and ChengXiang Zhai. A comparative study of methods for estimating query language models with pseudo feedback. In *Proceeding of the 18th ACM Conference on Information and Knowledge Management - CIKM '09*, pages 1895–1898, New York, NY, USA, 2009. ACM. 6.2.4.6

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *An Introduction to Information Retrieval*. Cambridge University Press, 2008. 1, 2.6.1, 6.2.1

Ryan McDonald, Koby Crammer, and Fernando Pereira. Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics - ACL '05*, pages 91–98, Morristown, NJ, USA, 2005. Association for Computational Linguistics. 5.3.3

Qiaozhu Mei, Hui Fang, and Chengxiang Zhai. A study of Poisson query generation model for information retrieval. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '07*, pages 319–326, 2007. 6.1.2.1

Donald Metzler. Generalized inverse document frequency. In *Proceeding of the 17th ACM Conference on Information and Knowledge Management - CIKM '08*, pages 399–408, New York, New York, USA, 2008. ACM Press. 1, 2.3.2, 5.2.2.5, 5.3.2, 6.2.4.5, 8.2.1, 8.3.2, 9.2.1

Donald Metzler and W. Bruce Croft. Combining the language model and inference network approaches to retrieval. *Information Processing and Management*, 40(5):735–750, September 2004. 2.6.5, 8.4.5

Donald Metzler, Victor Lavrenko, and W. Bruce Croft. Formal multiple-Bernoulli models for language modeling. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '04*, pages 540–541, 2004. 6.1.2.1, 9.3.1

Donald Metzler, Trevor Strohman, Howard Turtle, and W. Bruce Croft. Indri at TREC 2004: Terabyte track. In *Proceedings of 13th Text REtrieval Conference - TREC '04*. National Institute of Standard Technology, 2005. 6.2.4.6

Douglas P. Metzler, Terry Noreault, L. Richey, and B. Heidorn. Dependency parsing for information retrieval. In *Proceedings of the 7th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '84*, pages 313–324, Swinton, UK, UK, 1984. British Computer Society. 5.2.2.4

Mandar Mitra, Amit Singhal, and Chris Buckley. Improving automatic query expansion. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '98*, pages 206–214, 1998. 2.6.3, 2.6.3.2, 2.7.1.1, 8.2.2

Jae Hyun Park and W. Bruce Croft. Query term ranking based on dependency parsing of verbose queries. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '10*, pages 829–830, New York, NY, USA, 2010. ACM. 5.2.2.4

Fuchun Peng, Nawaaz Ahmed, Xin Li, and Yumao Lu. Context sensitive stemming for Web search. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '07*, pages 639–646, New York, New York, USA, 2007. ACM Press. 2.6.2, 8.2.2

Jeremy Pickens, Matthew Cooper, and Gene Golovchinsky. Reverted indexing for feedback and expansion. In *Proceedings of the 19th ACM Conference on Information and Knowledge Management - CIKM '10*, pages 1049–1058. ACM, 2010. 5.3.3

Jay M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '98*, pages 275–281, 1998. 1.1, 2.1, 6.1.2, 6.1.2.1

M. F. Porter. An algorithm for suffix stripping. In *Readings in Information Retrieval*, pages 313–316. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1997. 2.6.2

S. E. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-3. In *Proceedings of the Third Text REtrieval Conference - TREC '94*, pages 109–126. National Institute of Standard Technology, 1995. 1.1, 2.2.5, 6.1.1

Stephen E. Robertson and Karen Spärck Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27(3):129–146, May 1976. 1.4, 2.1, 2.2.1, 2.2.1, 2.2.2, 2.2.3, 2.3.2, 2.6.3, 2.6.5, 3.6, 5.4, 5.4, 6.1.2.2, 6.2.3, 7.2.2.3, 8.4.4, 9.2.1

Stephen E. Robertson and Stephen G. Walker. Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR' 94*, pages 232–241. Springer-Verlag, 1994. 2.2.5, 6.1.1, 6.2.3.2

Joseph John Rocchio. *Relevance Feedback in Information Retrieval*, pages 313–323. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1971. 2.1, 2.6.3, 2.6.3.1, 4.3, 9.3.2

G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18 (11):613–620, November 1975. 2.6.5

Gerard Salton, Edward A. Fox, and Harry Wu. Extended boolean information retrieval. *Communications of ACM*, 26:1022–1036, November 1983. 2.6.5, 8.4.5

Falk Scholer, Hugh E. Williams, and Andrew Turpin. Query association surrogates for Web search: Research articles. *Journal of the American Society for Information Science and Technology*, 55(7): 637–650, 2004. 2.6.1

Hinrich Schütze, David A. Hull, and Jan O. Pedersen. A comparison of classifiers and document representations for the routing problem. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '95*, pages 229–237. ACM Press, 1995. 2.6.6.1, 5.2.2

Amit Singhal. Modern information retrieval: a brief overview. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 24, 2001. 1

Alan F. Smeaton. Using NLP or NLP resources for information retrieval tasks. In *Natural Language Information Retrieval*, pages 99–111. Kluwer Academic Publishers, 1997. 2.7.3

Mark D. Smucker, James Allan, and Ben Carterette. A comparison of statistical significance tests for information retrieval evaluation. In *Proceedings of the 16th ACM Conference on Information and Knowlwdge Management - CIKM '07*, pages 623–632, 2007. 6.2.2

Karen Spärck-Jones, Stephen E. Robertson, Djoerd Hiemstra, and Hugo Zaragoza. Language modelling and relevance. *Language Modeling for Information Retrieval*, pages 57–71, 2003. 2.2.6

Trevor Strohman, Donald Metzler, Howard Turtle, and W. Bruce Croft. Indri: a language-model based search engine for complex queries. In *Proceedings of the International Conference on Intelligent Analysis*, 2005. 2.2.6, 2.6.3.2

Don R. Swanson. An introduction to medline searching. *Report, University of Chicago*, 2003. 2.7.2

Stephen Tomlinson. Experiments with the negotiated boolean queries of the TREC 2006 Legal Discovery track. In *Proceedings of the 15th Text Retrieval Conference - TREC '06*. National Institute of Standard Technology, 2007. 8.4.5

Stephen Tomlinson, Douglas W. Oard, Jason R. Baron, and Paul Thompson. Overview of the TREC 2007 Legal track. In *Proceedings of the 16th Text Retrieval Conference - TREC '07*. National Institute of Standard Technology, 2008. (document), 2.6.3.2, 8.2.2, 8.4.3.1, 8.4.6, 8.3

Elizabeth Tudhope. *Query based stemming*. PhD thesis, University of Waterloo, 1996. 2.6.2, 8.2.2

Howard Turtle. Natural language vs. boolean query evaluation: a comparison of retrieval performance. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '94, pages 212–220, New York, NY, USA, 1994. Springer-Verlag New York, Inc. 2.6.5

Howard Turtle and W. Bruce Croft. Inference networks for document retrieval. In *Proceedings of the 13st*

*Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '90*, pages 1–24, New York, New York, USA, 1990. ACM Press. 2.3.1, 2.6.5

C. J. van Rijsbergen. *Information Retrieval*. Butterworths, London, 2nd edition, 1979. 2.2.3

Ellen M. Voorhees. Natural language processing and information retrieval. In *Information Extraction: Towards Scalable, Adaptable Systems*, pages 32–48, London, UK, 1999. Springer-Verlag. 2.7.3

Xuanhui Wang and ChengXiang Zhai. Mining term association patterns from search logs for effective query reformulation. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management - CIKM '08*, pages 479–488, New York, New York, USA, 2008. ACM Press. 2.6.4, 8.2.2

Ryen W. White, Ian Ruthven, Joemon M. Jose, and C. J. Van Rijsbergen. Evaluating implicit feedback models using searcher simulations. *ACM Trans. Inf. Syst.*, 23:325–361, July 2005. 8.2.3

William A. Woods, Lawrence A. Bookman, Ann Houston, Robert J. Kuhns, Paul Martin, and Stephen Green. Linguistic knowledge can improve information retrieval. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, pages 262–267, Morristown, NJ, USA, 2000. Association for Computational Linguistics. 2.7.3

Jinxi Xu and W. Bruce Croft. Query expansion using local and global document analysis. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '96*, pages 4–11, 1996. 2.6.3

Xiaobing Xue, W. Bruce Croft, and David A. Smith. Modeling reformulation using passage analysis. In *Proceedings of the 19th ACM Conference on Information and Knowledge Management - CIKM '10*, New York, NY, USA, 2010. ACM Press. 8.2.2

Yiming Yang and Christopher G. Chute. An application of least squares fit mapping to text information retrieval. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '93, pages 281–290, New York, NY, USA, 1993. ACM. 2.6.6.3

C. T. Yu, K. Lam, and G. Salton. Term weighting in information retrieval using the term precision model. *Journal of the ACM*, 29(1):152–170, January 1982. 2

Chengxiang Zhai and John Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '01*, pages 334–342, New York, New York, USA, 2001. ACM Press. 1.1, 2.1, 6.1.2, 6.1.2.1

Le Zhao and Jamie Callan. Effective and efficient structured retrieval. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management - CIKM '09*, pages 1573–1576, New York, NY, USA, 2009. ACM Press. 2.7.1.2, 8.2.2, 8.4.5, 9.2.1

Le Zhao and Jamie Callan. Term necessity prediction. In *Proceedings of the 19th ACM Conference on Information and Knowledge Management - CIKM '10*, New York, NY, USA, 2010. ACM Press. 1.5, 2.5, 2.7.3, 5.2.2.1, 5.2.2.3, 5.2.2.7, 1, 8.2.1, 8.3.2

Le Zhao and Jamie Callan. The query dependent variation of term recall and its application in efficient term recall prediction (*submitted*). In *The 21st ACM International Conference on Information and Knowledge Management - CIKM '12*, 2012a. 1.5

Le Zhao and Jamie Callan. Automatic term mismatch diagnosis for selective query expansion (*accepted*). In *Proceedings of the 35th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '12*, 2012b. 1.5

Jianhan Zhu, Marc Eisenstadt, Dawei Song, and Chris Denham. Exploiting semantic association to answer 'vague queries'. In *Proceedings of the Fourth International Conference on Active Media Technology - AMT '06*, Brisbane, Australia, june 2006. 2.7.3

Yangbo Zhu, Le Zhao, Jamie Callan, and Jaime Carbonell. Structured queries for legal search. In *Proceedings of the 16th Text REtrieval Conference - TREC '07*. National Institute of Standard Technology, 2008. 2.6.3.2, 2, 8.4.3.1