TOWARDS BETTER SYSTEMS FOR COMPLEX SEARCH TASKS

by

Ashraf Bah

A dissertation submitted to the Faculty of the University of Delaware in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computer Science

Fall 2016

TOWARDS BETTER SYSTEMS FOR COMPLEX SEARCH TASKS

by

Ashraf Bah

Approved: _____
Kathleen McCoy, Ph.D.
Chair of the Department of Computer and Information Sciences

Approved: _____
Babatunde Ogunnaike, Ph.D.
Dean of the College of Engineering

Approved: _____
Ann L. Ardis, Ph.D.
Senior Vice Provost for Graduate and Professional Education

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Benjamin A. Carterette, Ph.D.
Professor in charge of dissertation

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Kathleen McCoy, Ph.D.
Member of dissertation committee

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Hui Fang, Ph.D.
Member of dissertation committee

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Yunyao Li, Ph.D.
Member of dissertation committee

ACKNOWLEDGMENTS

I am deeply grateful to my advisor, Benjamin Carterette, for his guidance and the invaluable time he devoted to me during this PhD journey. His thoughtful suggestions, recommendations and feedback about my ideas and all my research endeavors have helped shape me into the better researcher that I am today. I am particularly thankful for the freedom he gave me to explore interesting topics and ideas and to consequently grow as a researcher.

I am thankful to my committee members Kathleen McCoy, Hui Fang and Yunyao Li for their invaluable time and their feedback throughout the making of this thesis.

I would also like to thank my mentors from IBM, Yunyao Li, Howard Ho, Huaiyu Zhu and Prithviraj Sen and Jan Vondrack, from all of whom I was fortunate enough to learn a lot, and get different perspectives.

Many thanks to my current and former lab-mates Praveen Chandar, Mustafa Zengin, Karankumar Sabhnani, Mohammad Alsulmi, Ashwani Rao, Dongqing Zhu, Ivanka Zhuo Li, Rashida Davis and Priscilla Moraes with whom I shared not only lab spaces, but also ideas.

Finally, last but certainly not least, I would like to extend my gratitude to my parents for being my role models and for a job well done raising, guiding and supporting me. My gratitude also goes to my siblings and all my family members for their love and undying support.

TABLE OF CONTENTS

LIST OF TABLES x
LIST OF FIGURES xiii
ABSTRACT xv

Chapter

1 INTRODUCTION 1

1.1 Understanding Characteristics That Make Documents Desirable to Users in the Context of Broad Queries 4
1.2 Using Search History to Improve Current Search Results 4
1.3 Effective and Robust Models for Search and Retrieval 5
1.4 General Theoretical Framework for Retrieval and Ranking for Ad-Hoc, Diversity Ranking and Search over Sessions 6
1.5 Simulation of Query Reformulations 6

2 RELATED WORK 8

2.1 Traditional Ad-hoc Information Retrieval (IR) 8

2.1.1 Data Fusion 11
2.1.2 Novelty and Diversity Search and Retrieval 13
2.1.3 Risk-Sensitive Ranking 14

2.2 Search over Sessions 16
2.3 Simulation of Search Interactions 17
2.4 Text Retrieval Conference (TREC), NII Testbeds and Community for Information access Research (NTCIR) and ClueWeb corpora 19
2.5 Our Contributions 20

3 UNDERSTANDING USERS' PREFERENCES IN COMPLEX SEARCH TASKS 22

3.1 Preference Judgments 23

3.1.1 Dataset 24
3.1.2 Experimental Design 25

3.2 Comprehensiveness Analysis and Results 27

3.2.1 Questions Set 1 (Q1 through Q3) 29
3.2.2 Questions Set 2 (Q4 through Q6) 30
3.2.3 Questions Set 3 (Q7 through Q9) 32
3.2.4 Question Q10 33

3.3 Summary 35

4 USING DATA FUSION TO LEVERAGE SEARCH HISTORY IN ORDER TO IMPROVE CURRENT SEARCH RESULTS 36

4.1 Retrieval Data and Task 37
4.2 Retrieval Model 38

4.2.1 Collecting Related Queries 38
4.2.2 Aggregation Algorithms 41

4.3 Experiments and Results 41

4.3.1 Baselines: Language Model (LM) and Pseudo-Relevance Feedback 41
4.3.2 Evaluation Measures 42
4.3.3 Results 42

4.4 Summary 51

5 EFFECTIVE AND ROBUST MODELS FOR SEARCH AND RETRIEVAL 52

5.1 Retrieval task and Data 53
5.2 Retrieval Model 54

5.2.1 Estimating Document “Retrievability” 55
5.2.2 Reranking 56

5.2.2.1 Method 1 56
5.2.2.2 Method 2 57

5.3 Evaluation Measures 59
5.4 Experiments and Results 61

5.4.1 Effectiveness 61
5.4.2 Risk analysis 63

5.5 Summary 67

6 GENERAL THEORETICAL FRAMEWORK FOR RETRIEVAL AND RANKING FOR SEARCH OVER SESSIONS, AD-HOC AND DIVERSITY RANKING 68

6.1 Model 70

6.1.1 Model components 73

6.1.1.1 Query sample space Q 73
6.1.1.2 Query sample probability P (q'|$\mathcal{Q}$) 74
6.1.1.3 Probability of relevance P(rel|q', D) 75
6.1.1.4 Probability of retrieval P(retr|q', D) 76

6.1.2 Connection to data fusion methods 76

6.1.2.1 CombSUM 77
6.1.2.2 CombWSUM 77
6.1.2.3 CombMNZ 78

6.2 Experiments and Results 78

6.2.1 Data 79
6.2.2 Evaluation measures 80
6.2.3 Baselines 80
6.2.4 System Implementations 80

6.3 Results 82

6.3.1 Results for TREC Sessions 82

6.3.1.1 Choice of possible queries $\mathcal{Q}$ 82
6.3.1.2 Choice of P(retr|q', D) 83
6.3.1.3 Choice of P (q'|$\mathcal{Q}$, D) 84
6.3.1.4 Additional Analysis: Effects on Task Types 86
6.3.1.5 Additional Analysis: More about query sample space $\mathcal{Q}$ 87

6.3.2 Results for TREC Web 90

6.3.2.1 Choice of possible queries $\mathcal{Q}$ 91
6.3.2.2 Choice of P (retr|q', D) 92
6.3.2.3 Choice of P (q'|$\mathcal{Q}$, D) 92

6.3.3 Results for iMine 2014 93

6.4 Summary 93

7 SIMULATION OF SEARCH INTERACTIONS 95

7.1 Methodology 96

7.1.1 A simple model for generating search history data 96
7.1.2 A Somewhat More Complex Model 97

7.2 Implementation of the Methods 98

7.2.1 Layer 1 simulated query reformulations 99
7.2.2 Layer 2 simulated query reformulations 99

7.3 Experiments and Results 100

7.3.1 Dataset and Evaluation Measures 100
7.3.2 Effectiveness of Simulated Queries: Leveraging Simulated Queries 100
7.3.3 Results 101

7.3.3.1 Can we improve effectiveness at all by simulating queries and leveraging them? 101
7.3.3.2 When we aggregate the ranking resulting from the user's query with the rankings resulting from simulated queries, does it affect the results? 102
7.3.3.3 Does concatenating the simulated queries with the original query impact the results? 103
7.3.3.4 Is there any added value in going down to layer 2 and deeper? 103

7.4 Summary of initial simulation attempt 104
7.5 Pilot Study: Towards more human-like simulations of query reformulations 104

7.5.1 Framework and Data 104
7.5.2 Model 105

7.5.2.1 Topical Language Model 106
7.5.2.2 Sampling Queries 108
7.5.2.3 Scoring Sample Queries 109
7.5.2.4 Session abandonment 110

7.5.2.5 Summary of the model 110
7.5.3 Experiments and Results 111
7.6 Summary 116
8 CONCLUSION AND FUTURE WORK 118
REFERENCES 123

LIST OF TABLES

Table 3.1: Results for all the questions, all results in the last column are significant (++) at p<0.01, except for Q5 and Q9 28

Table 4.1: Example of related queries obtained using Xtrak4Me method for the query “cosmetic laser treatment” 40

Table 4.2: nDCG@10 using session-dependent data 44

Table 4.3: α-nDCG@10 for 2011 using session-dependent data. 45

Table 4.4: nDCG@10 and ERR@10 using Bing and combinations of Bing and session-dependent data 46

Table 4.5: α-nDCG@10 and ERR-IA@10 using Bing and combinations Bing and session-dependent data. 46

Table 4.6: nDCG@10 for query expansion + CombMNZ 46

Table 4.7: nDCG@10 for different cutoffs for top documents 47

Table 4.8: nDCG@10 for various ways of exploiting click data 47

Table 4.9: 2011 Bing+Xtrak4Me between top-10 docs and the docs from previous interactions (Best of bing+session specific) 48

Table 4.10: nDCG@10 per task type for 2012 session using session-dependent data for CombMNZ 49

Table 4.11: nDCG@10 per task type for 2012 session using Bing alone and combinations of Bing and session-dependent data for CombMNZ. B is short for Bing 50

Table 5.1: TREC 2013 results for Method2. Bold font denotes positive difference between Method2 and the submitted-run, in terms of ERR-IA@20. + denotes statistical significance 61

Table 5.2: Summary of effectiveness. This shows, for each measure for each dataset, the number of runs for which a given method performs better (on average) than another method – and vice versa. Alg1 stands for method1. Base is short for baseline 62

Table 5.3: Summary of Risk-sensitive results (U-ERR). This shows, for each measure for each dataset and with respect to a specific baseline, the number of runs for which a given method is more robust than another method – and vice versa. α=5 64

Table 5.4: Risk sensitive measures using indri as baseline for each run on 2013 datasets. "Alg1.UERR@20" is the risk-sensitive measure, using Method1. "Alg1-base.UERR@20" is the difference between Method1 and the submitted-run, in terms of U_{RISK}(ERR@20). α=5 64

Table 5.5: TREC 2013 results for Method1. Bold font denotes positive difference between Method2 and the submitted-run, in terms of ERR-IA@20. + denotes statistical significance 66

Table 6.1: TREC Session track results for different combinations of model components. * indicates a statistically significant difference over the baseline by a paired t-test at the 0.05 level. The biggest improvement for each dataset is bolded. 85

Table 6.2: TREC Session track results broken out by goal and product types. 86

Table 6.3: TREC Session track results for different sources of possible queries $\mathcal{Q}$. * indicates a statistically significant difference over the baseline by a paired t-test at the 0.05 level. The biggest improvement for each dataset is bolded. 87

Table 6.4: TREC Web track results for different combinations of model components. * indicates a statistically significant difference over the baseline by a paired t-test at the 0.05 level. The biggest improvement in each column is bolded. For the best reported TREC result, we report the highest value of the measure across all submissions; the system with the highest nDCG is not necessarily the same as the system with the highest α-nDCG. 91

Table 6.5: NTCIR iMine 2014 results using the query suggestions provided by the organizers of the task. 93

Table 7.1: Results for layer 1 of both models on Session track 2013 dataset. Q0x, Q1x, Q2x, Q3x and Q4x respectively denote incorporating the real user query ranking 0, 1, 2, 3 and 4 times in the set of rankings that are being aggregated. QN denotes the concatenation of the real user query to the simulated query. 101

Table 7.2: Comparing Layer1 to "Simpler L1+L2" as well as "Complex L1+L2" 103

Table 7.3: Top 4 most-frequent terms appearing in queries for three topics with their binomial occurrence model probability P ($w \in Q$), their multinomial language model probability P ($w|Q$), and their frequency in queries sampled using the procedure in Section 7.5.2.2. 108

Table 7.4: Overall precision@10 averaged across all sessions and all rounds in each session for each of our six systems. 113

Table 7.5: Examples of sequences of queries for six topics in our set. For each topic we show an actual user session of queries and a simulated session 115

LIST OF FIGURES

Figure 1.1: Example of queries for completing the complex search task about "Vacation in Sydney" 2

Figure 3.1: Screenshot of the HIT Layout 24

Figure 3.2: Comparison of proportions in which questions are true/false for three cases (considering all-pref, left-pref only and right-pref only). 34

Figure 4.1: nDCG@10 using CombCAT, CombSUM and CombMNZ on Session track 2012 data 43

Figure 4.2: ERR@10 using CombCAT, CombSUM and CombMNZ on Session track 2012 data 44

Figure 4.3: Difference between Xtrak4Me and LM in terms of the number of documents that cover 2 or more aspects (y-axis) for all sessions, on the 2011 dataset. 51

Figure 4.4: Number of tokens introduced by Alch, Xtrak, Snip and Bing that are not part of the main query, but are part of the topic description on the 2011 dataset. 51

Figure 5.1: Algorithm 1 56

Figure 5.2: Illustration of Algorithm 1 57

Figure 5.3: Algorithm 2 58

Figure 5.4: Illustration of Algorithm 2 59

Figure 7.1: A somewhat simple model for generating search history data: The elliptical shapes with large dashes represent the generated simulated queries 97

Figure 7.2: A somewhat more complex model for simulating session search data: The elliptical shapes with large dashes represent the generated simulated queries. The elliptical shapes with dashes represent the keyphrases generated in the new steps (they can also be used as simulated queries)..... 98

Figure 7.3: Comparison of six retrieval systems across six rounds of a session using non-simulated queries..... 112

Figure 7.4: Comparison of six retrieval systems across six rounds of a session using simulated queries..... 112

ABSTRACT

When users interact with search engines, in a large number of cases, they first formulate a query and examine the results, and then reformulate it one or several more times until they either satisfy their information need or give up. Complex search tasks fall into those cases. Unlike simpler tasks in which users are looking for a particular homepage or a particular single piece of information or an answer to a single specific question, complex search tasks often span multiple search queries (i.e. a sequence of queries) and can span multiple sessions (i.e. multiple sequences of queries).

In this thesis, we present several efforts for building more effective and robust retrieval systems for complex search tasks in situations where we have only small amounts of search history data. We first start by investigating and understanding users' preferences with respect to document comprehensiveness and topical relevance grade.

Then, using our findings from that experiment, we introduce heuristic data fusion methods to improve search results in a search session by leveraging most recent search history and query logs.

Next, we go beyond simple average effectiveness by considering risk-sensitivity as an essential part of our retrieval systems. For that purpose, we present re-ranking approaches that exploit the "popularity" of documents and we show that they produce results with improved robustness and effectiveness over a variety of retrieval systems used as baselines. Risk-sensitive ranking (or robustness-aware ranking) focuses on improving the robustness of the system by minimizing the risk of

obtaining, for any topic, a result subpar with that of the baseline system. In other words, robustness refers to the ability of the ranker to reduce and mitigate poor performance on certain individual queries while striving to improve the overall performance as well.

Our next endeavor consists in going beyond heuristic retrieval models. For that purpose, we propose a probabilistic data fusion framework for retrieval and ranking inspired by the well-known probability ranking function, and we use it to solve search over sessions, as well as ad-hoc search, novelty and diversity search.

Finally, in order to achieve high effectiveness for search over a session even in the absence of search history, we propose to simulate search interactions that provide data similar to what we could have obtained if a user were to have prior interactions with the search engine (previous queries, top results returned for previous queries, etc.).

Chapter 1

INTRODUCTION

Information Retrieval (IR) is concerned with finding documents or resources relevant to a user's information need. IR encompasses, among other things, activities such as representation, storage, retrieval, evaluation, and ranking of documents, web pages, images and videos. At a high level, our efforts revolve around the retrieval of documents and the user modeling that it entails. More specifically, our focus is on proposing retrieval models aimed at improving users' satisfaction during complex search tasks and search over sessions, even when there is only little search history data available.

Complex search tasks are tasks that are characterized by users interacting with a search engine in a session: formulating a query, viewing some results, reformulating based on what they see in those results or the idea of the task they have in mind, drifting to related topics, and so on. We define them in contrast to simple tasks such as homepage finding or factual question answering or finding one highly-relevant piece of information, all of which are common tasks that search engines tend to be very good at already.

Let us consider vacation planning as an example. A user may enter the following search query: "vacation in Sydney." That user may be interested in finding more information about some or all of the following aspects: "cheap flights to Sydney", "Sydney vacation package deals", "top reasons to vacation in Sydney", "best things to do in Sydney", "cheap hotels in Sydney" and "car rental in Sydney." Clearly

satisfying several of these aspects of the user's information need is likely to lead the user to engage in many interactions with the search engine, and to provide several reformulations of the query. Figure 1.1 illustrates all the possible transitions between these queries. In this case, a user can potentially start with any query and proceed with any other query from the list, and so on – and possibly re-issuing the same query again.

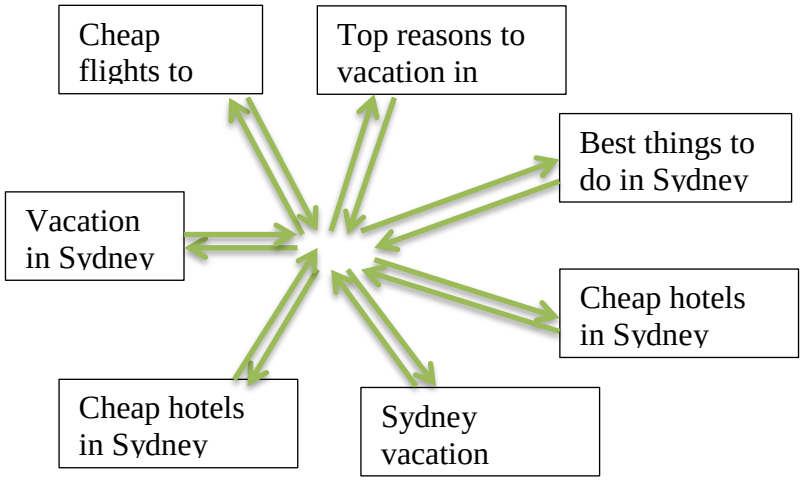


Figure 1.1: Example of queries for completing the complex search task about "Vacation in Sydney"

In our thesis, we conduct several studies aimed at the creation of information retrieval systems with strong performance for complex search tasks in situations where we have only very small amounts of search history data. First of all, we believe that with complex search tasks – and with many IR tasks in general –, the user is central. Therefore, it is important to understand users and their preferences in complex search tasks and subsequently build systems that account for those preferences. For that

reason, we propose to perform a user study to understand users' preferences in the context of complex search tasks.

Secondly, given the interactive nature of complex search tasks, systems that are built to perform retrieval for complex search tasks must be able to leverage users' previous interactions with the search engine in the search session to provide more effective results. For that purpose, we propose heuristic data fusion models that leverage users' search history in order to improve search effectiveness.

With complex search task systems as well as with many IR systems, it is common place for researchers and developers to focus on search effectiveness and completely ignore robustness. In our work, we advocate to take robustness into consideration and thus propose heuristic robustness-aware reranking models.

Given that models that are more grounded in well-studied theories have the advantage of being better understood and more solid, we propose a general theoretical framework for retrieval as an improvement to our heuristic retrieval models. For that, we introduce a probabilistic data fusion framework for retrieval and ranking for various retrieval tasks including ad-hoc, diversity and session search.

Finally, in the cases of cold-start in a complex search where there is no search history data to leverage, we believe it will be beneficial to generate pseudo search history data. Therefore, we propose ways to generate pseudo search history data by simulating query reformulations.

The steps we took are described below.

1.1 Understanding Characteristics That Make Documents Desirable to Users in the Context of Broad Queries

In order to understand users' preferences in complex search task scenarios, we can study a similar but simpler task such as novelty and diversity search task which, unlike complex search tasks, assumes there is only one query formulation. In both scenarios, users are interested in seeing documents that are not only relevant, but also cover more aspects (or subtopics) related to the topic of interest. For the purpose of understanding users' preferences, we propose in Chapter 3 to conduct a user study where users are asked to prefer one of two documents B and C given a broad query and also given that they have already seen a document A. We then answer several questions pertaining to the relationship between the "comprehensiveness" of documents (i.e. the number of subtopics a document is relevant to) and real users' preference judgments. Using this user study framework, we were able to empirically show that users strongly prefer documents with high subtopic coverage (i.e. more comprehensive documents), regardless of the topical relevance grade.

1.2 Using Search History to Improve Current Search Results

Based on the results of the study described in the previous section, we propose a retrieval model that tends to prioritize documents that are likely to be relevant to the largest number of subtopics; for each document, this can be naively gauged by considering the number of related queries whose rankings a document appears in.

On the other hand, it has been observed that during their interactions with search engines, many users find themselves reformulating their queries in order to satisfy their information need, thus engaging into many interactions (i.e. sequences of user queries and resulting search engine rankings) over the same search session. Hence, we decided to move away from the traditional assumption that each query is

independent, and apply our retrieval model in a context where there are previous search interactions – and consequently search history from those interactions. The retrieval model we propose is a heuristic data fusion approach that leverage users' search history in this context of search over sessions. In Chapter 4, we describe the retrieval data we used in this context of search over sessions, as well as the heuristic data fusion method we used for leveraging users' search history, and the results we obtained.

1.3 Effective and Robust Models for Search and Retrieval

Although effectiveness is the main and most important measure of the performance of information retrieval systems, it is also important to ensure that the systems are robust. Robustness refers to the ability of the ranker to reduce and mitigate poor performance on individual queries while striving to improve the overall performance as well. Robust systems mitigate situations where the improved systems fare worse than the baseline on certain queries, even though the average effectiveness score is higher than the baseline's. In order to create robustness-aware effective systems, we propose two re-ranking approaches based on exploiting the popularity of documents with respect to a general topic. We used each of the runs (i.e. ranked list of documents) submitted to a popular information retrieval evaluation conference as baseline, and empirically show that our algorithms improve the effectiveness as well as the robustness of the systems in an overwhelming number of cases, even though the systems used to produce the runs employ a variety of retrieval models. In Chapter 5, we describe the dataset we used for risk-sensitive ranking (i.e. robustness-aware ranking) as well as the approaches we used to improve robustness of the baseline systems along with the results we obtained.

1.4 General Theoretical Framework for Retrieval and Ranking for Ad-Hoc, Diversity Ranking and Search over Sessions

The techniques we mentioned in Section 1.2 and detailed in Chapter 4 are somewhat ad hoc and lacking in underlying principles that might explain their efficacy. In order to get a more principled effective retrieval method, we propose a framework based on a modified version of the well-known Probability Ranking Principle [84]. Instead of following the guideline that says that optimal ranking is achieved when documents are ranked in decreasing order of probability of relevance, we make an amendment as follows: optimal ranking is achieved when documents are ranked in decreasing order of probability of relevance and "retrievability". We argue that if we can obtain a set of "possible queries" for an information need, and compute the probability of retrievability of documents with respect to each query as well as the probability of relevance of each of the documents, we can achieve high effectiveness by fusing the rankings over the possible queries. In Chapter 6, we describe in more detail the principled approach that we propose to follow.

1.5 Simulation of Query Reformulations

In search scenarios where we are in the presence of very little to no search history data, we propose to simulate query reformulations that provide data similar to what we can obtain when a user has previous interactions with the search engine (previous queries, top results returned for previous queries, etc.). Specifically, assuming that we have a real user who provides one single query and nothing else, we address the question of whether we can generate data that can be considered to be similar to search history data, and that leads to results similar to the ones we obtain when we leverage real users' search history. We also investigate whether we can improve search effectiveness by leveraging simulated queries, and how that would

compare to leveraging real search history. More details on how we simulate query reformulations are presented in Chapter 7.

Chapter 2

RELATED WORK

2.1 Traditional Ad-hoc Information Retrieval (IR)

Ad-hoc IR has long focused on improving average overall effectiveness, and treats queries as though they are single faceted. Popular state-of-the-art retrieval models that have been used in ad-hoc IR include language modeling [81] and Okapi BM25 [86]. The Markov Random Field model for term dependencies has been proposed by Metzler and Croft [75]. And so were other term proximity models such as those described by Buttcher et al. [25] and Tao and Zhai [95]. In recent years, learning to rank algorithms have been adopted [24, 69], as well as learning to re-rank algorithms [58], novelty and diversity ranking [26] and risk-sensitive ranking [100].

In the following lines, we briefly describe some of the most popular relevance models.

- **Boolean Retrieval Model**

This is the first and simplest retrieval model. It is based on Boolean logic and classical set theory. The user's query is conceived as a set of terms combined by Boolean operators AND, OR, NOT, etc. Each retrieved document in the returned unordered list of documents contains those query terms [23].

- **Vector Space Model**

This model represents documents or queries as a vector of term weights. Each term is part of the vocabulary in question, and the relevance score is often computed using cosine similarity. The most well-known function for obtaining the term weights

is TF-IDF [87]. TF (term frequency) is the raw frequency of a term within a document [71] while IDF (inverse document frequency) quantifies whether the term is common or rare across all documents [92].

$$IDF(t) = \log \frac{N}{df_t}$$

where N is the total number of documents in the collection C, and df_t is the number of documents in C that contain the term t.

- **Okapi BM25**

BM25 is a TF-IDF-like ranking function based on a probabilistic retrieval framework introduced by Robertson and Jones.

$$score(D,Q) = \sum_{i=1}^{n} IDF(q_i) \cdot \frac{tf_{qi,D} \cdot (k_1 + 1)}{tf_{qi,D} + k_1 \cdot (1 - b + b \cdot \frac{|D|}{avgdl}}$$

where |D| is the length of the document D in words, avgdl is the average document length in the collection, k_1 and b are free parameters, and IDF(q_i) is often defined as:

$$IDF(q_i) = \log \frac{N - n(q_i) + 0.5}{n(q_i) + 0.5}$$

where N is the total number of documents in the collection, and $n(q_i)$ is the number of documents containing q_i.

- **Query Likelihood Model and the Language Model family**

Language modeling [20, 52, 76, 81] is a probability distribution over words. Query likelihood is part of that family of retrieval models. Each document is assigned a score which is the probability of the document being relevant given a query.

$$score(D,Q) = \log P(Q|D) = \sum_{i=1}^{n} log \frac{tf_{qi,D} + \mu \frac{tf_{qi,C}}{|C|}}{|D| + \mu}$$

where qi is the ith term in query Q, n is the total number of terms in Q, |D| and |C| are the document and collection lengths in words respectively, $tf_{qi,D}$ and $tf_{qi,C}$ are the document and collection term frequencies of qi respectively, and μ is the Dirichlet smoothing parameter [81]. Smoothing is a common technique to estimate the probability of unseen words in the documents [37, 80, 109, 110]. Notice that query terms are assumed to be generated independently from the language model.

- **Relevance Model and the Mixture of Relevance Model**

Relevance Model [64] is an example of document likelihood model – as opposed to query likelihood model – wherein we leverage the set of pseudo-relevant documents to obtain more text to use in the estimation of the language model.

$$P(w|\hat{\theta}_Q) \propto \frac{1}{|R|}\sum_{D\epsilon R} P(w|\theta_D)P(Q|\theta_D)$$

where R is the set of pseudo-relevant document, θ_D is a document language model, $P(Q|\theta_D)$ is the query likelihood and w is a possible query term.

Mixture of Relevance Model is an improvement over relevance modeling wherein information in external document collections is leveraged [45]. This has been shown to achieve more stable MAP improvement than traditional pseudo-relevance feedback across a range of news and web collections. The term generation probability function is formally defined by:

$$P(w|\hat{\theta}_Q) = \sum_{C_i} kC_i \frac{P(C_i)}{|R_{Ci}|}\sum_{D\epsilon R_{C_i}} P(w|\theta_D)P(Q|\theta_D)$$

where kCi is the normalization factor for the relevance model estimate using collection Ci.

- **Markov Random Field (MRF), Weighted Sequential Model (WSD) and Positional Language Model (PLM)**

In reality the independence assumption in the Query Likelihood model is rather naive, since related terms are likely to appear in close proximity to each other.

The Markov Random Field (MRF) model [75] improves it by accounting for term proximity. It works by first constructing a graph that contains a document node, one node (i.e. random variable) per query term, and edges that define independence semantics between the random variables. Then, it models the joint distribution over the document random variable and query term random variables.

$$P_\Lambda(Q|D) = \sum_{c \in T} \lambda_T f_T(c) + \sum_{c \in O} \lambda_O f_O(c) + \sum_{c \in U} \lambda_U f_U(c)$$

where T is the set of 2-cliques containing the document node and a query term node, O is the set of cliques involving the document node and two or more query terms that appear contiguously in the query, and U is the set of cliques involving the document node and two or more query terms that appear non-contiguously within the query. f(c) is the feature function over clique c and λ's are the feature weights.

The Weighted Sequential Model (WSD) is an extension of the MRF model wherein the query concept weights – specifically the lambda parameters of the previous equation – are automatically learned [19].

Like MRF and WSD, the positional language model (PLM) [72] accounts for term proximity. The PLM is estimated for each position based on propagated counts of words within a document through a proximity-based density function. The document relevance score is calculated by scores of its PLMs. PLM can be further improved by incorporating pseudo-relevance feedback [73].

2.1.1 Data Fusion

Previous research has shown that combining the evidence of multiple query representations helps improve results in information retrieval [Belkin96]. Two such

methods are CombSUM and CombMNZ. For each query, CombSUM reranks documents based on their cumulative scores over all the various query representations (i.e. related queries). CombMNZ, on the other hand, uses the sum of similarity values times the number of non-zero similarity values [67].

In general, we can fuse rankings from different systems [49, 90] or rankings obtained using different document representations in order to improve performance.

While some fusion techniques use the scores of the documents across the ranked lists [14, 97], others use their ranks instead [48, 66], and some others use their probability of occurring in a predefined segment of the lists [65, 68]. Bartell et al. propose a linear combination method by which the relevance estimates made by rankers can be automatically combined for better performance [14]. Aslam and Montague [4] propose to use a metasearch model based on Borda Count and another one based on Bayesian Count and show that both usually outperform the best input system and are competitive with, and often outperform, existing metasearch strategies. Montague and Aslam [78] also propose to apply ideas from the Social Choice Theory, specifically they employ the Condorcet procedure for data fusion and show it to outperform the model based on Borda Count. Lillis et al. propose a method that assumes that the performance of the individual input systems on a number of training queries is indicative of their future performance, and thus fuses the rankings based on probabilities of relevance calculated during this training process [68]. In another effort, Wu and McClean propose a group of algorithms to eliminate the effect of uneven correlation among component results by assigning different weights to all component results or their combinations. They then use the linear combination method or a variation to fuse the results [102].

In most cases, using data fusion in IR helps obtain good results. However, that positive impact depends on the quality of the input result lists [48, 96, 79, 103]. Thus an essential part of our work is dedicated to finding appropriate sources of data to be used as various query representations.

2.1.2 Novelty and Diversity Search and Retrieval

Another aspect of ad-hoc retrieval is focused on improving diversity and novelty search and retrieval. The goal of diversity and novelty retrieval search and retrieval is to provide rankings where, not only is document relevance accounted for, but also users' intents as well as the different aspects of the topic being searched on. Many of the diversity ranking approaches are inspired by the MMR algorithm introduced by Carbonnell and Goldstein [Carbonnell98]. The fundamental idea of MMR is to optimize for both document relevance and coverage of intents and/or aspects. Differences in implementations lie in how similarities are computed: Carbonnell and Goldstein suggest using any similarity function such as cosine similarity. Zhai et al. [108] advocate for modifying the language modeling framework to incorporate a model of relevance and redundancy. Other researchers utilized the correlation between documents as a measure of their similarity in the pursuit of diversification and risk minimization in document ranking [100]. Carterette and Chandar [27] introduced a greedy result set pruning wherein there are 2 steps: in the first step, they rank documents in decreasing order of their similarity to the query; and in the second step, they proceed to iteratively prune documents whose similarity to any previously selected document is greater than a certain threshold. They also use a set-based probabilistic model to maximize the likelihood of covering all the aspects [27]. Radlinski and Dumais [82] exploited a commercial search engine to obtain

aspects of queries, and proceeded to diversify the ranking using query-query reformulations. Santos et al. [88] utilize a query-driven approach wherein they explicitly account for aspects by using sub-queries to represent a query. They then estimate the relevance of each retrieved document to every identified sub-query, and the importance of each sub-query.

2.1.3 Risk-Sensitive Ranking

Risk-sensitive ranking (or robustness-aware ranking) focuses on improving the robustness of the system by minimizing the risk of obtaining, *for any topic*, a result subpar with that of the baseline system. Most research in Information Retrieval (IR) has focused on improving the average effectiveness of systems. However, it is very often the case that the improved systems fare worse than the baseline on many of the queries, even though the average effectiveness score is higher than the baseline's. The concept of robust ranking appears therefore to be key, when it comes to remedying those cases. In this present work, robustness refers to the ability of the ranker to reduce and mitigate poor performance on individual queries while striving to improve the overall performance as well.

Research on risk-sensitive ranking is still in its infancy. Perhaps the work most related to our approach is Wang et al.'s effort [100] to address robustness by proposing a principled learning to rank framework that optimizes for both effectiveness and robustness of a retrieval system. Essentially the authors proceeded by proposing a learning method that optimizes for both reward and risk with respect to a given baseline. While their approach consists in learning to control the tradeoff between effectiveness and robustness, our approach is a general re-ranking method that can be used on top of any retrieval model (e.g. language model, query expansion,

learning to rank, data fusion, or Markov Random Field retrieval models) and that consists in re-arranging documents in the original ranking, in order to get a more effective and robust final ranking.

Another major contribution in the literature is that of Dincer et al. [46] that focused on uncovering biases inherent to the way TREC Web track evaluated systems prior to 2014. The authors argued that, given there were several various retrieval methods used by the track participants in building the systems (query expansion, learning to rank, etc.), comparing robustness using one single baseline that was created with a specific retrieval model creates inherent biases. That is, systems that build on top of retrieval models similar to the baseline will have an advantage over others, and systems that build on top of retrieval models very different from the baseline will be at great disadvantage. The authors proposed several ways to mitigate that issue including the use of mean within-topic system effectiveness as a baseline. In the present work, our focus is not to show that we can improve robustness with respect to one single specific baseline. Rather we focus on showing that, given any ranking obtained using a certain retrieval model, we can apply our method to improve the robustness as well as the overall effectiveness of the system. For that purpose, we used each and every run from TREC Web 2013 and 2014 – as baselines – to empirically show that.

Wang and Zhu [99] proposed a risk-aware ad-hoc retrieval model that utilized the correlation between documents as a measure of their similarity in the pursuit of diversification and risk minimization in document ranking. In a similar effort to establish a risk-aware framework for retrieval, Zhu et al. [112] proposed to model uncertainty and utilized a one-parameter loss function to model the level of risk acceptable by a user. Their loss function was applied to a language modeling

framework. Our approach, however, is a general re-ranking approach that we apply to the ranking of any retrieval model, and that we show to improve average overall effectiveness as well as risk-sensitive measures. There are other efforts for ad-hoc risk-aware retrieval methods that focus on query expansion cases [40] and pseudo-relevance feedback cases [74].

It is to be noted however that the term robustness as used in our study differs from the sense it is given in other work like in Bhattacharjee's work wherein robust ranking means ranking algorithm that is sensitive (or less vulnerable) to spams and noise in the training set [21].

2.2 Search over Sessions

A session is a sequence of query reformulations and interactions made by a user in order to satisfy an information need. A session consists of a current query and the previous interactions that led to the current query. In many search scenarios, users tend to provide one or more reformulations of their original query in order to satisfy their information need. The TREC Session track was created to investigate such use cases. The main goal of that track is to improve the ranking of a current query given information about the previous interactions (document Id, title, URL, snippets, clicked URLs and dwell time of previous queries) [59, 61]. The track provides test collections that contain this data.

Several approaches have been proposed. Zhang et al. [111] proposed to tackle the problem using a relevance feedback model that takes advantage of query changes in a session. In Zhang's work, when computing the relevance score between a current query and a document, terms' weights are increased or decreased depending on whether the term was added, retained or removed from previous queries in the session.

In a similar approach [51], Guan et al. proposed to model sessions as Markov Decision Processes wherein the user agent's actions correspond to query changes, and the search agent's actions correspond to increasing or decreasing or maintaining term weights.

Raman et al. [83] used related queries to diversify results while maintaining cohesion with the current query in order to satisfy the current query as well as future queries. They did so using a two-level dynamic ranking where the user's interaction in the first level is used to infer her intent in order to provide a better ranking that includes second level rankings.

Query aggregation has also been used by Guan who identified text nuggets deemed interesting in each query and merges them to form a structured query [50]. Notable approaches used by TREC participants include the work of Jiang et al. [55] in which the authors combined Sequential Dependence Model features in both current queries and previous queries in the session for one system, and combined that method with pseudo-relevance feedback for other systems. Another notable approach is the use of anchor texts for query expansion proposed by Kruschwitz et al. [63] and adopted by others.

2.3 Simulation of Search Interactions

Keskustalo et al. [62] were among the first to argue for test collections that include a model of user query reformulation for the purpose of evaluation over sessions. They propose a test collection consisting of multiple query reformulations created by selecting terms from a manually-defined space. Reformulations are generated according to one of four simulation models derived from observing real users' search strategies. They define session success as finding one relevant document,

and failure as finding none. With that definition, they could evaluate their simulation models/search strategies in terms of the total time (or cost) required to achieve success. They conclude by arguing that IR test collections should "model processes where the searcher may try out several queries for one topic"; the present work is an answer to that.

In follow-up studies, Baskaya et al. [17 and 15] propose a query modification simulation strategy to model how words are selected to form an initial simulated query or subsequent queries. They start with a fixed set of terms for each topic, then perform all possible permutations and focus on five strategies that users employ for query modification. For the purpose of simplicity, the model intentionally disregards the fact that users may learn by reading snippets and inspecting documents. Baskaya et al. also [16] model scenarios in which the user involved in a search process based on relevance feedback can make mistakes by providing erroneous feedback. They showed that errors generated under their simulation models can actually provide results as good as correct relevance feedback.

There has been other work on simulating queries and aspects of user behavior. Van Dang and Croft [44] showed that anchor text could be used to generate query reformulations. Jiang et al. [55] propose a model for how users browse results over a session in order to evaluate reformulations. Carterette et al. [30] sample values of parameters in parametrized evaluation measures like RBP to simulate sampling users for an interactive evaluation. In separate works, Yang and Lad [106] and Kanoulas et al. [59a] define session evaluation measures based on an expected utility model of user browsing over a session; Jarvelin et al. [54] defined a session evaluation measure based on DCG.

2.4 Text Retrieval Conference (TREC), NII Testbeds and Community for Information access Research (NTCIR) and ClueWeb corpora

TREC is an annual evaluation conference organized by the National Institute of Standards and Technology. The goal is to strive to standardize experimental settings and datasets for researchers and developers who are working on some search and retrieval tasks. The tasks can vary from year to year, and each year there are several organized tasks. In general, for each task, organizers provide information needs as well as collections of documents to participants (researchers and developers). Each participant will thus create a search system that performs the same task for the track they are participating in, and all participants will use the same dataset. Afterwards, TREC organizers collect and make available human relevance judgments as well as other tools and resources necessary for measuring and understanding the effectiveness of the search systems. TREC datasets that are of interest to us for our thesis are the 2011 and 2012 Session track datasets [60, 61] which we describe in detail in Chapter 4, the 2013 and 2014 Session track datasets [32, 31] which we describe in detail in Chapter 6, and the novelty and diversity and risk-sensitive ranking track datasets (TREC Web track 2013 and 2014 datasets) [41], [42] – which we describe in detail in Chapters 5 and 6.

NII Testbeds and Community for Information access Research (NTCIR) is another evaluation conference with similar goals as TREC. However NTCIR is organized by a different entity, the National Institute of Informatics (NII) in Japan. The tracks/tasks organized by NII are not the same as TREC's. But some of the tasks share similar subtasks. Relevant to our thesis is the iMine task which, like the TREC 2013 and 2014 Web tracks, is also a diversification task for IR. More details about the iMine dataset is provided in Chapter 6.

For all seven datasets - TREC Sessions track 2011, 2012, 2013 and 2014, TREC Web track 2013 and 2014, NTCIR IMine 2014, the corpus used for retrieval was either ClueWeb09 or ClueWeb12. ClueWeb09 corpus is a set of about 1 billion web pages in ten languages that were crawled during the months of January and February 2009. ClueWeb12 corpus is a set of of 733,019,372 English web pages that were crawled between February 10, 2012 and May 10, 2012.

The corpus on which the retrieval is performed for the 2011 and 2012 Session track tasks is the category B subset of ClueWeb09. The category B subset contains 50 million documents. For the 2013 and 2014 TREC Web and Session tracks, the retrieval corpus is the entire ClueWeb12 corpus. For the NTCIR IMine task, the retrieval corpus is the ClueWeb12-B13 subset which contains about 50 million documents.

2.5 Our Contributions

Our work builds on top of some of these retrieval models in a few ways. Before delving into our proposed retrieval models, we present in Chapter 3 our work on understanding users' preferences in the context of complex search tasks. We view this study as motivating our models: it suggests that users value novelty and diversity of relevant content (i.e. document comprehensiveness) over the simple presence of relevant information. Thus the models we build aim to fuse information from diverse—but related—sources in order to provide users with the most variety of relevant information in their search results.

In Chapter 4, we employ a heuristic data fusion technique to leverage users' search history in the context of search over sessions. First, for each given query, we generate related queries from various sources and use those multiple representations of

a query to obtain several rankings. Then we combine those multiple rankings using simple rank aggregation methods. We then proceed to show that our results are very effective.

In Chapter 5, we go beyond average effectiveness. We propose two reranking approaches – based on exploiting the "popularity" of documents with respect to a general topic – that improve the effectiveness while improving the robustness of the baseline systems. We used several systems as baselines and show that, even though these were systems based on a variety of retrieval models, our approaches increase their robustness.

In Chapter 6, we go beyond simple heuristic data fusion methods by proposing a principled probabilistic data fusion framework for retrieval and ranking that effectively addresses search over sessions as well as ad-hoc search, and novelty and diversity search. Our framework is based partially on the well-known Probability Ranking Principle, which states that the optimal ranking of documents is achieved when sorting them in decreasing order of probability of relevance [84]. We augment this principle with the following guideline: optimal ranking is achieved when documents are ranked in decreasing order of probability of relevance and "*retrievability*," instead of relevance only.

Finally, in Chapter 7, we propose to simulate query reformulations that provide data similar to what we can obtain when a user has previous interactions with the search engine. This will be considerably helpful in cases where there is very little to no actual search history data to leverage. Chapter 8 concludes our work.

Chapter 3

UNDERSTANDING USERS' PREFERENCES IN COMPLEX SEARCH TASKS

Users with complex search tasks often start with very broad queries that may span several aspects of a topic. They may want documents that are not only relevant to their information need, but also cover a diverse set of aspects of their information need. We are interested in their preferences for relevance, diversity, and novelty in the sense of a document covering an aspect that is not covered by a previously-ranked document. It is essential to understand real users, and use our insights about their preferences to inform our retrieval methods for complex search tasks.

For the purpose of understanding users' preferences, we conduct a pilot user study where users are asked to assign a preference to one of two documents B and C given a broad query and also given that they have already seen a document A. We then investigate a total of ten questions pertaining to the relationship between the "comprehensiveness" of documents (i.e. the number of subtopics a document is relevant to) and real users' preference judgments. The main research questions we answer are: Are users inclined to prefer documents with higher comprehensiveness, even when the prior document A already covers more aspects than the two documents being compared, and even when the less preferred document has a higher relevance grade? Furthermore, are users inclined to prefer documents with higher overall aspect-coverage even in cases where B and C are relevant to the same number of novel subtopics. We propose to adopt this user-study framework – the so-called triplet

framework – because previous studies have shown it to be a good framework for collecting judgments for novelty based on preferences [Chandar and Carterette 2012, Chandar and Carterette 2012b]. In fact, it was previously used by Chandar and Carterette to test different sets of hypotheses pertaining to novelty ranking tasks. We regard the present work, which is more focused on document comprehensiveness, as a complement to their work.

3.1 Preference Judgments

Absolute judgments in IR such as Boolean relevance and graded relevance have been used widely in the literature. An alternative is pairwise preference judgments, in which an assessor is presented with two documents and gives a preference to one document over the other. Early work on preference judgments in IR involved inferring preferences from those absolute judgments [24]. However more recently, pairwise (i.e. binary) preference judgments have been adopted as an alternative that may offer advantages in terms of alleviating the burden on assessors by reducing the complexity of the assessment, rendering the assessment task easier than assigning grades and reducing assessor disagreement [29].

The preference judgment scheme we adopt here, however, is a bit different and was first proposed by Chandar et al. [33]. This scheme is based on the so-called triplet framework, wherein given a query and a document A that the assessor is to pretend contains everything they know about the topics, the assessor chooses the *next* document they would like to see between two documents.

3.1.1 Dataset

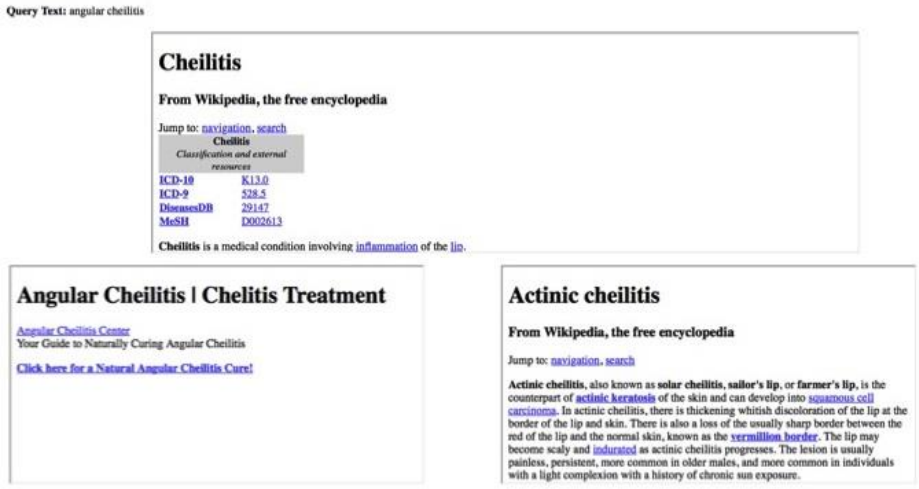


Figure 3.1: Screenshot of the HIT Layout

For our experiments, we use 10 topics from the TREC 2012 Web track dataset [38]. The 10 topics selected are a sample of *broad* topics (i.e. topics good for *intrinsic diversity* tasks as well as *complex search* tasks). Documents were selected for preference assessment from those judged for the Web track. All documents are therefore from the ClueWeb09 collection of millions of web pages. We use the publicly available subtopic relevance judgments produced by experienced NIST assessors and based on graded relevance. Since topical relevance of documents was not explicitly provided, we consider the maximum subtopic relevance grade of a document to be its topical relevance (ad-hoc relevance). For each one of the 10 topics, we obtained triplets from several users. In the next section, we describe the experimental design that yielded the triplets.

3.1.2 Experimental Design

The framework we adopt in this pilot study for preference judgment is based on the work of Chandar and Carterette [33]. In the framework, an assessor (i.e. user) is shown three documents, one appearing at the top of the page, one appearing at the bottom left, and the third appearing at the bottom right. We will refer to the top document as D_T, and the bottom ones as D_1 and D_2, in concordance with naming conventions of Chandar and Carterette. Given D_T and a topic, the assessor is asked to choose between D_1 and D_2. Essentially the assessor would need to indicate their preference for the *second* document they would like to see in a ranking by selecting either D_1 or D_2.

Since we have graded topical and subtopic relevance judgments for the documents, we can use that information to determine what subtopics each document is relevant to, as well as the corresponding relevance grades. A document can thus be represented as the set of subtopics it has been judged relevant to. For instance, $D_i = \{S_j, S_k\}$ means document i is relevant to subtopics j and k.

We used Amazon Mechanical Turk (AMT) [3] – an online labor marketplace – to collect user judgments. AMT works as follows: a requestor creates a group of Human Intelligence Task (HITs) with various constraints and workers from the marketplace work to complete the tasks. Workers were instructed to assume that everything they know about the topic is in the top document, and they are now trying to find a document that would be most useful for learning more about the topic. No mentions of subtopics, novelty, or redundancy were given to them except as examples of properties assessors might take into account in their preferences (along with recency, ease of reading, and relevance). Each preference triplet consists of three documents, all of which were relevant to the topic; the documents were picked

randomly from the data described in the previous section. One document appeared at the top followed by two documents below it. The HITs layout design, the quality control decisions and the HIT properties were the same as described by Carterette and Chandar [34]. Figure 3.1 shows an example of a triplet as it appeared in a HIT.

In online crowdsourcing marketplaces like AMT, due to the subjective nature of this task, it is primordial to ensure that the workers are taking their task seriously, and not just randomly clicking for the sake of making quick money. We have a few ways of weeding out poor quality preference judgments: Majority vote, trap questions and qualifications.

Majority vote: In order to ensure that the workers are not randomly clicking around and that they actually understand the task, one simple yet commonly used quality check is to have many workers perform the same HIT and then use some way to determine the one(s) that are off. We use majority vote in this case. Each HIT was completed by five different workers, and the document that the majority of them agree to be preferable is the one we choose to be the preferred one.

Trap Questions: Another way to avoid poor quality judgments is by slipping trap questions in the midst of questions. Essentially, trap questions are questions for which we already knew the answers, and which we use to eliminate poor quality workers. Specifically, we have "non-relevant document traps" and "identical document traps". In "non-relevant document trap", one of the documents is not relevant to the topic at all, so there is no way a good quality worker would prefer it over the relevant document. In "identical document trap," although the two documents being compared are relevant to the topic, one of them is identical to the prior (already seen) document. So, there is no reason for a good quality worker to prefer the identical

document over the non-identical relevant one that contains new information. Out of the five triplets in each HIT, one contains a trap question that is randomly selected.

Qualifications: On Amazon Mechanical Turk, task requestors can see workers' historical performance (i.e. ratings) on other previous tasks they performed, and they can decide and set the acceptable threshold performance that workers must have before getting involved in the tasks. We used two qualifications in our experiment. We allowed only workers that have an approval rate of 95% or more. This is an easy way to eliminate spammers and ill-intentioned workers who randomly click around. The other qualification technique we employed was a simple test – the so-called qualification test. The goal is to ensure that users actually understand the task that they will be performing. In our study, we need users to understand what it means to select a document with novel information, given the prior known document. For that purpose, we initially present the workers with a HIT layout that looks the same as the ones they will be using if qualified. On that HIT, there is one "non-relevant document trap" and two "identical document traps." Workers have to correctly complete that task before they can proceed.

3.2 Comprehensiveness Analysis and Results

In this section, we enumerate some specific questions about relationships between document "comprehensiveness", ad-hoc (i.e. topical) relevance grade, and user preferences. Our goal is to test the degree to which comprehensiveness (in the sense of covering more aspects in relevance judgments) is more important to users than topical relevance grade; in general, we hypothesize that it is the more important of the two factors in their preferences.

Table 3.1: Results for all the questions, all results in the last column are significant (++) at p<0.01, except for Q5 and Q9

	T 152	T 157	T 158	T 167	T171	T 173	T178	T 184	T 196	T199	All Topics
Q1 true/false	196/ 22	87/ 13	148/19	97/ 28	181/31	155/30	119/51	157/19	108/53	136/22	1384/288 (82.78% true)++
Q2 true/false	0/0	0/0	4/1	10/2	5/0	0/0	1/1	4/4	0/0	6/1	30/9 (76.92% true)++
Q3 true/false	10/15	0/0	33/13	18/6	12/1	6/11	14/11	21/14	18/15	12/14	144/90 (61.54% true)++
Q4 true/false	40/3	20/2	49/14	31/6	42/15	82/21	37/17	56/8	59/32	41/10	457/128 (78.12% true)++
Q5 true/false	0/0	0/0	0/0	0/0	0/0	0/0	1/1	2/2	0/0	6/1	9/4 (69.23% true)
Q6 true/false	4/1	0/0	21/4	0/0	0/0	2/7	6/1	1/6	5/2	8/6	47/27 (63.51% true)++
Q7 true/false	181/21	67/11	97/5	79/24	149/18	73/9	105/50	115/11	49/21	91/11	1006/181 (84.75% true)++
Q8 true/false	0/0	0/0	4/2	10/10	5/0	0/0	1/1	2/2	0/0	0/0	22/7 (75.86% true)++
Q9 true/false	10/5	0/0	9/24	18/6	12/1	4/4	13/17	16/8	13/13	4/8	99/86 (53.51% true)
Q10 true/false	15/1	20/2	63/18	20/5	32/13	84/28	18/5	49/16	64/34	59/18	424/140 (75.18% true)++
All Q's T/F	293/68	194/28	400/100	283/87	306/79	406/110	315/155	423/90	316/170	363/91	3622/960++

In the following, $D_1 > D_2$ means document D_1 is preferred to document D_2. $R_1 > R_2$ means document D_1 was judged by NIST assessors to have a higher ad-hoc relevance grade than document D_2. $S_1 > S_2$ means document D_1 contains more subtopics than document D_2. $S_1^{new} > S_2^{new}$ means document D_1 contains more novel subtopics (with respect to the subtopics already seen in D_T) than document D_2. Regardless of whether a document was placed on the left or right of a triplet, we will refer to the more comprehensive one as D_1.

3.2.1 Questions Set 1 (Q1 through Q3)

This first set of questions asks whether a document D_1 with higher aspect coverage, in general, tends to be preferred by users – regardless of whether D_1 covers more novel aspects than the document it is being compared to. The three questions are:

Q_1: If $S_1 > S_2$ and $R_1 > R_2$ then is $D_1 > D_2$? This asks if users prefer a document with higher aspect coverage and higher ad-hoc relevance grade than a document with lower aspect coverage and lower ad-hoc relevance grade.

Q_2: If $S_1 > S_2$ and $R_1 < R_2$ then is $D_1 > D_2$? This asks whether users prefer a document with higher aspect coverage but lower ad-hoc relevance grade than a document with lower aspect coverage but higher ad-hoc relevance grade.

Q_3: If $S_1 > S_2$ and $R_1 = R_2$ then is $D_1 > D_2$? This asks if, for documents with equal ad-hoc relevance grade, users prefer a document with higher aspect coverage than a document with lower aspect coverage.

We expected Q_1 to be largely true, and Q_2 and Q_3 to be more mitigated (or possibly inconclusive for Q_2) due to the fact that relevance grades are an important factor as well. The results in Table 3.1 show that the answer is true for all three questions: a document D_1 with higher aspect coverage, in general, tends to be

preferred by users – regardless of whether D_1 covers more novel aspects than the document D_2 it is being compared to. In fact, even Q2 and Q3 are true far more often than we expected them to be. According to the results, when D_1 covers more aspects than D_2 and D_1 also has a higher ad-hoc relevance grade than D_2, D_1 was by far preferred by users. In our experiment this happened 1384 times (82.78%), and failed to happen 288 times (17.22%).

The results also confirm Q3 to be true, which means that, for documents with equal ad-hoc relevance grade, users prefer the document with higher aspect coverage. And this happened 144 times (61.54%), and failed to happen 90 times (38.46%). Q2 is also true more often than not, i.e. 30 times (76.92%) against 9 times (23.08%). The results, while proving Q2 and Q3 to be true, also suggest that when the least-comprehensive document has a higher relevance grade than the most-comprehensive document, the bias against the least-comprehensive document is reduced.

3.2.2 Questions Set 2 (Q4 through Q6)

The second set of questions zooms into special cases where the prior document D_T (i.e. document shown at the top) has higher aspect coverage than each of D_1 and D_2 and asks whether, even then, the document with higher aspect coverage is preferred by users. The three questions are:

Q_4: If $S_1 > S_2$ given that ($S_T > S_1$ and $S_T > S_2$) and $R_1 > R_2$ then is $D_1 > D_2$? This asks if, given the prior document has higher aspect coverage than each of D_1 and D_2, users still prefer a document with higher aspect coverage and higher ad-hoc relevance grade than a document with lower aspect coverage and lower ad-hoc relevance grade.

Q_5: If $S_1 > S_2$ given that ($S_T > S_1$ and $S_T > S_2$) and $R_1 < R_2$ then is $D_1 > D_2$? This asks if, given the prior document has higher aspect coverage than each of D_1 and D_2, users prefer a document with higher aspect coverage but lower ad-hoc relevance grade than a document with lower aspect coverage but higher ad-hoc relevance grade.

Q_6: If $S_1 > S_2$ given that ($S_T > S_1$ and $S_T > S_2$) and $R_1 = R_2$ then is $D_1 > D_2$? This asks if given the prior document has higher aspect coverage than each of D_1 and D_2, for documents with equal ad-hoc relevance grade, users prefer a document with higher aspect coverage than a document with lower aspect coverage.

The results in Table 3.1 show that the answer is yes for questions Q_4 through Q_6. That is, even when the prior document D_T has higher aspect coverage than each of D_1 and D_2, the document D_1 with higher aspect coverage is preferred by users. And here again, whether documents D_1 and D_2 have the same relevance grade or not, it is the one with the highest aspect coverage that gets selected by users as most preferred. Although, as we expected, Q_5 and Q_6 are more mitigated than Q_4. Q_4 is very often true (in 78.12% of qualifying user preferences): given D_T with higher aspect coverage than D_1 and D_2, the document D_1 with higher aspect coverage and higher ad-hoc relevance grade is preferred. Also, Q_5 is often true (69.23% of qualifying user preferences, but this is not significant): given D_T with higher aspect coverage than D_1 and D_2, the document D_1 with higher aspect coverage but lower ad-hoc relevance grade is preferred. And finally, H_6 is also often true (63.51% true and 36.49% false): given D_T with higher aspect coverage than D_1 and D_2, the document D_1 with higher aspect coverage but the same topical relevance grade as the other document, is preferred. However, the proportions in which Q_5 and Q_6 are true are not as strong as that of Q_4,

which suggests that the bias against the least-comprehensive document is reduced in the cases of Q_5 and Q_6.

3.2.3 Questions Set 3 (Q7 through Q9)

This third set of questions focuses on "novel-comprehensiveness". Given a prior document D_T, we say that a document D_1 is more "novel-comprehensive" than D_2 if D_1 covers more novel subtopics (with respect to the subtopics already seen in D_T). Here we investigate whether a document D_1 with higher novel aspect coverage, in general, tends to be preferred by users. The three questions are:

Q_7: If $S_1^{new} > S_2^{new}$ and $R_1 > R_2$ then is $D_1 > D_2$? This asks whether users prefer a document with higher novel-aspect coverage and higher ad-hoc relevance grade than a document with lower novel-aspect coverage and lower ad-hoc relevance grade.

Q_8: If $S_1^{new} > S_2^{new}$ and $R_1 < R_2$ then is $D_1 > D_2$? This asks if users prefer a document with higher novel-aspect coverage but lower ad-hoc relevance grade than a document with lower novel-aspect coverage but higher ad-hoc relevance grade.

Q_9: If $S_1^{new} > S_2^{new}$ and $R_1 = R_2$ then is $D_1 > D_2$? This asks if, for documents with equal ad-hoc relevance grade, users prefer a document with higher novel-aspect coverage than a document with lower novel-aspect coverage.

The results shown in Table 3.1 show that the answer to questions Q_7 through Q_9 is yes. In fact, Q_7 is true in 1006 cases (84.75%), and fails 181 times (15.25%). This means when D_1 covers more novel aspects than D_2 and D_1 also has a higher ad-hoc relevance grade than D_2, D_1 was by far preferred by users. Q_9 is true in 99 cases (53.51%), and fails 86 times (46.49%). This means when D_1 covers more novel aspects than D_2 but has the same ad-hoc relevance grade as D_2, D_1 was still selected by users over D_2 in more cases, but the gap is not significant. Also, Q_8 is true only

slightly more often than it is false. It is true 22 times (75.86%), and false 7 times (24.14%). These results suggest that when the least novel-comprehensive document has less than or equal ad-hoc relevance grade as the most novel-comprehensive document, the bias towards the most novel-comprehensive document is reduced.

3.2.4 Question Q10

This final question puts an emphasis on cases where the two documents being compared are equally "novel-comprehensive" – i.e. cover the same number of new subtopics – and posits that even in that case, users are more likely to prefer the one that covers the most number of subtopics.

Q_{10}: If $S_1 > S_2$ given that ($S_1^{new} = S_2^{new}$) then is $D_1 > D_2$? This asks whether users are more likely to prefer a document that covers the most number of subtopics, even when both documents contain the same number of novel-aspects. In other words, users are biased towards more comprehensive documents, even in cases where both documents have the same number of novel-aspects.

The result for this question is perhaps the most interesting one. It shows that, even when the two documents being compared are relevant to an equal number of novel aspects, users are more inclined to choose the one with the highest overall subtopic coverage. And this happens in 75.18% of cases (424 times out of 564).

It should be noted that there are very few cases (17 cases) where the preferred document covers more aspects but fewer novel-aspects; and even fewer cases (2 cases) where it contains more novel-aspects but fewer aspects.

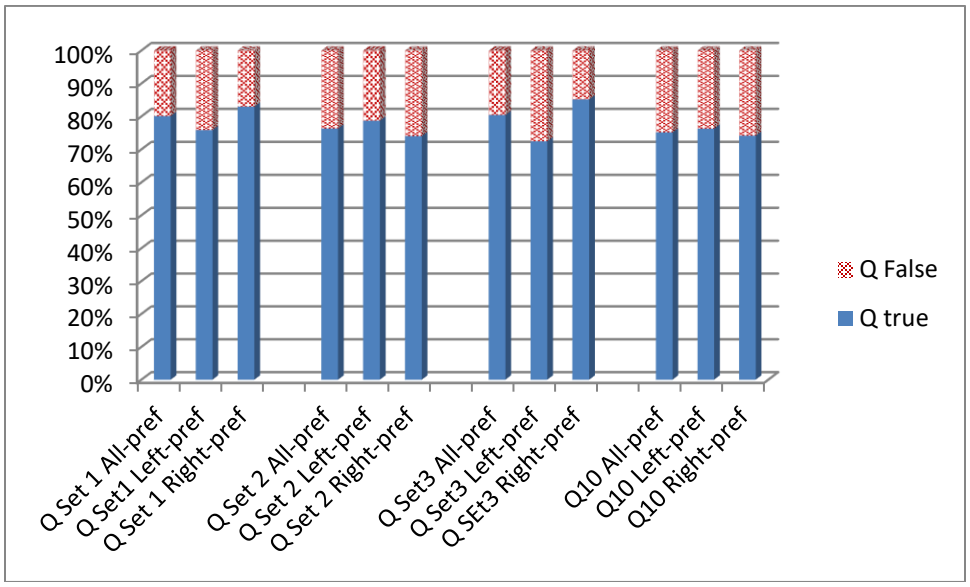


Figure 3.2: Comparison of proportions in which questions are true/false for three cases (considering all-pref, left-pref only and right-pref only).

It is important to note that, in most cases, the document with higher aspect coverage (i.e. more comprehensive) is either more relevant or equally relevant to the other document. There were not many cases where one of the document being compared is more comprehensive but with lower ad-hoc relevance. So those cases are underrepresented, possibly due to the fact that documents with high coverage tend to be very relevant.

But is it the case that users prefer left docs to right docs (or vice versa) even when the preferred document has lower aspect coverage? That is, does the position of the document have an effect on it being preferred by a user? The results in Figure 3.2 suggest that is not the case. In fact, the proportions in which the answers to the questions are true/false in both situations are relatively close. The triplets were indeed

placed randomly in either left or right, that is, they are not placed according to any factor.

3.3 Summary

In this chapter, we conducted a pilot study in which we have used the triplet framework to empirically show that users tend to prefer in large proportions documents with high aspect coverage, regardless of the topical relevance grade. We asked users to choose, given a prior document D_T, between two documents D_1 and D_2 the one that is most useful for learning more about the topic. According to the results, users overwhelmingly prefer documents that are relevant to the largest number of aspects (i.e. highest aspect coverage), even when the prior document D_T already covers more subtopics than each of D_1 and D_2. In fact, even in cases where D_1 and D_2 are relevant to the same number of novel subtopics, the one that is relevant to the largest overall subtopics tends to be preferred.

Chapter 4

USING DATA FUSION TO LEVERAGE SEARCH HISTORY IN ORDER TO IMPROVE CURRENT SEARCH RESULTS

From the previous chapter, we learned that users engaged in complex search tasks tend to prefer documents with high subtopic/aspect coverage, regardless of the topical relevance grade. And we can naively gauge this for each document by considering the number of related queries (i.e. queries similar to a user's query for the information need) whose rankings a document appears in. Hence we have decided to propose and use a data fusion method that promotes documents that are likely to cover high numbers of subtopics relevant to the information need. Data fusion for information retrieval consists in combining results from multiple different retrieval systems. The idea that retrieval could be improved by simply combining results from multiple different retrieval systems – a sort of "wisdom of the crowd" for retrieval systems – has long been attractive to IR researchers and practitioners. This idea produced much work in data fusion methodologies, which use different ways of combining retrieval scores from different ranking functions or different rankings of documents to produce a final ranking based on input from all available systems. In this chapter we present our work on using data fusion to leverage users' recent search history to improve their search results tackling the problem of search over sessions. First, we identify useful sources for terms to be used as related queries. For each query, we generate related queries from various sources and use those multiple representations of a query to obtain several rankings that we combine using simple

rank aggregation methods. We compare the effects of using each source and show that our simple heuristic method is effective.

4.1 Retrieval Data and Task

For our experiments, we use the TREC 2011 and 2012 Session track datasets [60, 61]. In each, there are several sessions containing one or many interactions. The 2011 dataset contains 76 sessions while the 2012 dataset contains 98 sessions. A session is a sequence of query reformulations and interactions made by a user in order to satisfy an information need. A session consists of a current query and the previous interactions that led to the current query. In our datasets, an interaction more specifically consists of a query and a ranked list of documents for the query, along with titles, URLs and query-biased snippets (produced by Yahoo! BOSS) for each document. Also included are clicked documents and the time spent by the user reading a clicked document. During the experiment that led to the collection of sessions and interactions by the TREC organizers, web pages shown to the users were retrieved using Yahoo! BOSS. Thus the retrieved results included pages that were not part of the retrieval corpora ClueWeb09; those pages were filtered out before results were displayed to the users.

Session track retrieval task consists in ranking results for the last query (current query) in the session using their systems. The corpus on which the retrieval is performed for the 2011 and 2012 dataset is the category B subset of ClueWeb09. This was the same (sub)set used by the top TREC Session track systems.

We indexed the collection using Indri, an open-source package for indexing and retrieval that implements the inference network model [94].

The 2011 dataset includes subtopic judgments, making it possible to compute diversity measures for the 2011 dataset.

4.2 Retrieval Model

There are two components to our approach. First, we generate and select related queries by leveraging various sources of information from the user's search history, and submit those related queries to the Indri search engine [94]. Finally, we aggregate the retrieved results for those queries. Note that for all our experiments in this chapter, we filter retrieved results using Waterloo's fusion spam classifier with a threshold of 0.75 [43] in an effort to exclude spams from our ranked lists.

4.2.1 Collecting Related Queries

Using Session-Dependent Data

The intuition behind using users' session data is that users may be reformulating their queries using pieces of information that were displayed to them during their previous interactions with the search engine [107, 44] and such pieces can give more insight about the user's actual intent, context or topic. Specifically, we use the following pieces of information as related queries or alternative queries that users could have provided to the search engine in order to satisfy their information need:

1. The previous queries in the user's session;
2. The titles of documents ranked for previous queries in the user's session;
3. Entire snippets (free of stop words) for the top 10 documents ranked for previous queries in the user's session;
4. Most significant key-phrases extracted from top-10 ranked documents for previous queries in the user's session. In this case we concatenate

the texts of the top 5 key-phrases from each document into one new single longer query;

Titles, snippets and concatenation of key-phrases can be considered to be alternative longer queries, more verbose than the usual queries provided by real users in most cases.

Key-phrases were extracted using a model that exploits linguistic and statistical methods. The method uses statistical lexical analysis to determine the most significant single-word terms, and extracts those terms as well as their immediate context to form complex terms. Then it proceeds by clustering similar complex terms – using Monge-Elkan distance [77] as the string-similarity measure – and selecting a representative for each cluster to be a candidate key-phrase. For selecting a representative, a similarity maximization algorithm is used that prefers the key-phrase that resembles the remaining key-phrases most closely. Finally, all the candidates are analyzed in order to determine confidence scores for each in the context of the document in question. The confidence scores are obtained by combining the significance of cue tokens in the representing candidate, the scope, as determined by the distribution of the candidate cluster over the document, and number of words contained in the candidate.

Xtrak4Me is an open-source library that implements and performs the key-phrase extraction described above [89]. We also use AlchemyAPI, a black box but production-ready key-phrase extraction tool, to ensure that similar results can be obtained using other key-phrase extraction algorithms [1]. And given the similarly good performances achieved by both the production-ready tool and the academic open-source method, it is fair to conclude that decent key-phrase extraction algorithms

can provide us with good enough key-phrases to be used as candidate related queries for the purpose of our experiment.

We submit the previous queries, titles, snippets, or key-phrases as queries to an Indri index of ClueWeb09 and aggregate results using the methods described below. Examples of related queries obtained from some documents in interaction 1 of session 7 in the 2011 dataset can be seen in Table 4.1 for Xtrak4Me method. The user's query for that interaction was "cosmetic laser treatment".

Table 4.1: Example of related queries obtained using Xtrak4Me method for the query "cosmetic laser treatment"

Xtrak4Me (expanded) from a document retrieved during search history	#weight(0.5 #combine(cosmetic laser treatment) 0.100 laser 0.100 skin 0.100 treatment 0.100 removal 0.100 acne)
Xtrak4Me from a document retrieved during search history	laser skin treatment removal acne
Xtrak4Me (expanded) from a document retrieved during search history	#weight(0.5 #combine(cosmetic laser treatment) 0.100 cynosure 0.100 practitioner 0.100 inc 0.100 removal 0.100 skin)
Xtrak4Me from a document retrieved during search history	cynosure practitioner inc removal skin

Using Bing's Related Queries

In this method, we use Bing query suggestion service to obtain related queries for each current query. We provide queries using Bing API and obtain query suggestions in response by the service [22]. We submit them as queries and fuse the results like in the previous case.

Using a combination of Bing related queries and session-dependent data

Here our aim is to observe what happens when we add Bing related queries to the set of related queries obtained from each source of session-dependent data.

4.2.2 Aggregation Algorithms

Previous research has shown that combining the evidence of multiple query reformulations helps improve results in information retrieval [18]. Two such methods are CombSUM and CombMNZ. For each query, CombSUM reranks documents based on their cumulative scores over all related queries. CombMNZ, on the other hand, uses the sum of similarity values times the number of non-zero similarity values [67]. Additionally, we propose another method, CombCAT with the purpose of investigating what happens when we give precedence to documents that appear the most in our rankings. In CombCAT, for each query, we first group documents into different categories such that documents that appeared in n different rankings are put in the same category labeled "category$_n$". Then we proceed with our re-ranking by promoting documents that appear in the largest number of rankings. And within the same category, documents that have the largest sum take priority over others. Both CombMNZ and CombCAT explicitly reward documents that appear in the largest number of rankings – though in different ways.

4.3 Experiments and Results

4.3.1 Baselines: Language Model (LM) and Pseudo-Relevance Feedback

The first baseline we are comparing our results to is obtained using Indri's language model (LM) on each of the current queries [94]. Essentially, for each session, we submit only the current query (i.e. the last query in the session) to Indri.

Then we filter these baseline results using Waterloo's fusion spam classifier with a threshold of 0.75 [43] to exclude spams.

For the second baseline, we use an adaptation of Lavrenko's pseudo-relevance feedback as implemented in Indri. We run the model with 6 different values for the number of documents used in the feedback: 10, 50, 100, 200, 300, 400 and 500. For each of the two datasets, we select the run with the maximum nDCG@10 as our second baseline. The best feedback size is 50.

4.3.2 Evaluation Measures

For traditional measures, we opted for using two of the measures adopted by the TREC Session track organizers, namely the primary measure nDCG@10 [53] and the second measure ERR@10 [Chappelle09]. Given that a user's information throughout a session might span several aspects of a topic, we deemed it necessary to use diversity measures to evaluate our methods. We used α-nDCG@10 [39] and ERR-IA@10 [Chappelle11] as diversity measures.

4.3.3 Results

In the following tables, B is short for Bing, Q for Query, Alch for AlchemyAPI, Xtrak for Xtrak4Me and Snip for Snippets. Since nDCG@10 is the official score used by TREC Session track organizers, we report nDCG@10 for traditional relevance measure and α-nDCG@10 for diversity-focused measure. nDCG@10 and ERR@10 results follow similar trends, and so do α-nDCG@10 and ERR-IA@10 results. The best nDCG@10 is 0.431 for 2011 (when using Bing+Xtrak, as can be seen in Table 4.4) and 0.314 for 2012 (as can be seen the query expansion experiment in Table 4.6). These results are on par with TREC Session track systems

for 2011 and 2012. To put things in context, in 2012, our best run would be squeezed between the best run of the top group and the best run of the second-best team (which had nDCG@10 of 0.3221 and 0.3033 respectively). Similarly, in 2011, our best run would be squeezed between the best run of the second-best group and the best run of the third-best team (which had nDCG@10 of 0.4409 and 0.4307 respectively, and were dominated by the top team which reached 0.4540).

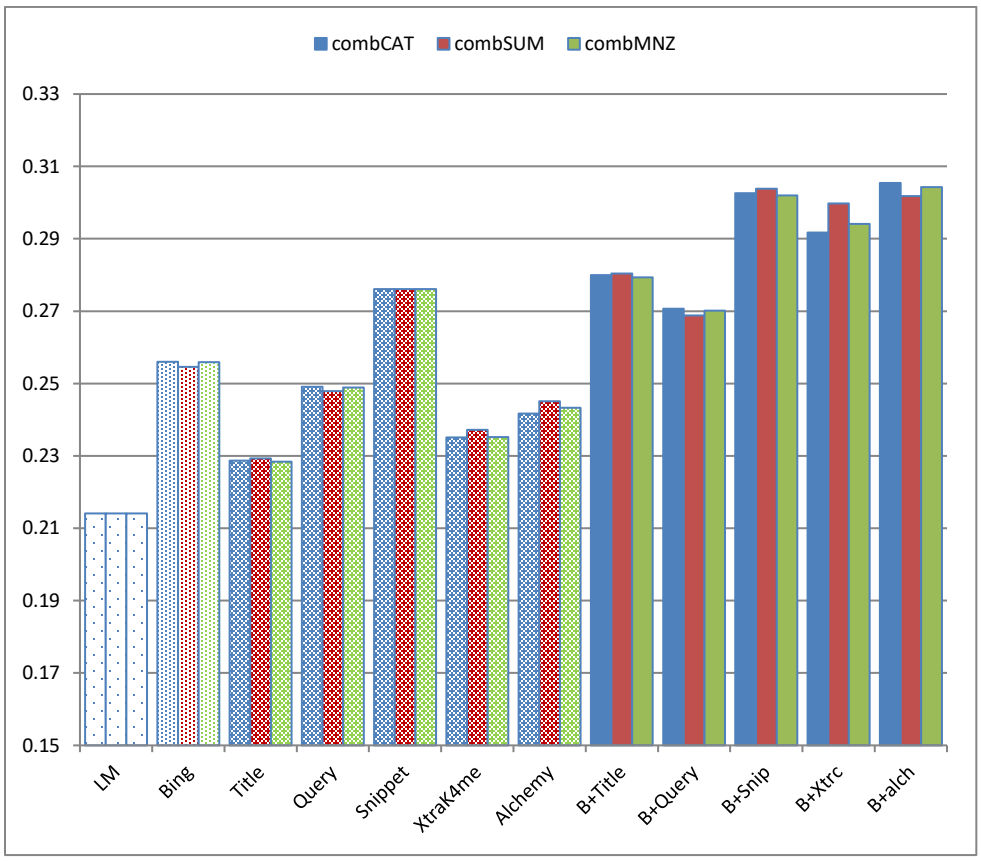


Figure 4.1: nDCG@10 using CombCAT, CombSUM and CombMNZ on Session track 2012 data

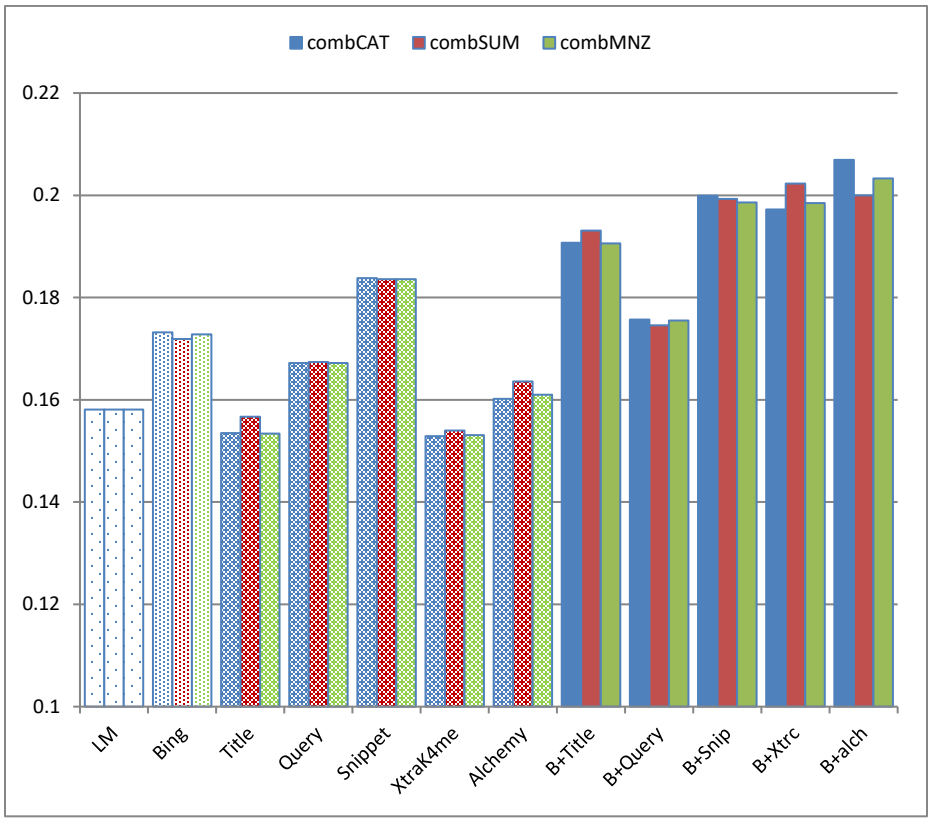


Figure 4.2: ERR@10 using CombCAT, CombSUM and CombMNZ on Session track 2012 data

Note: In our tables, + and ++ mean statistically significant at $p<0.05$ over the LM and pseudo-relevance feedback baselines respectively.

Table 4.2: nDCG@10 using session-dependent data

	LM	Query	Snippet	Xtrak4me	Alch
2012 ndcg@10	0.214	0.249	0.274	0.235	0.244
% increase	0%	16.49%	27.88%++	9.86%	13.87%
2011 ndcg@10	0.318	0.379	0.377	0.391	0.365
% increase	0%	19%	18%	23%+	15%

Using Bing's related queries only, we achieved a good improvement over the Indri baseline (19.06% nDCG@10 increase) for the 2012 dataset as shown in Table 4.4. The impact on the 2011 data using Bing only is more moderate (4% nDCG@10 increase). We get better improvements when we use session-dependent data, as can be seen in Table 4.2. In particular, for 2012, we get a peak of 27.88% increase when we use snippets as related queries, and for 2011, we get a peak of 23% increase when we use Xtrak4Me key-phrases.

Combining Bing's related queries with session-dependent data gives even better results, as shown in Table 4.4. For the 2012 data, using nDCG@10, we observe a peak of 42.5% increase when we use B+Alch and a low of 26.16% increase when we use B+Q. In fact for both 2011 and 2012 datasets, combing Bing and any session-dependent data produces a better result than using that session-dependent data alone. This may be because the combinations cover more aspects than the range of aspects covered by each source's queries alone. ERR@10 results follow the same trend as nDCG@10.

Table 4.3: α-nDCG@10 for 2011 using session-dependent data.

	LM	Query	Snippet	XtraK4me	Alchemy
2011 α-ndcg@10	0.374	0.435	0.476	0.474	0.426
% increase	0%	16.28%+	27.22%+	26.90%+	13.85%

The 2011 session test set includes relevance judgments for subtopics as well. Using that, we were able to compute α-nDCG@10 and ERR-IA@10 diversity measures for the 2011 dataset. The trend is similar to that of traditional measures, with a peak of 44.72% increase for ERR-IA@10 (and 36.98% α-nDCG@10 increase) when combining Bing and XtraK4Me. The results can be seen in Table 4.5.

Table 4.4: nDCG@10 and ERR@10 using Bing and combinations of Bing and session-dependent data

	LM	Bing	B+Q	B+Snip	B+Xtrak	B+alch
2012 ndcg@10	0.214	0.255	0.270	0.299	0.295	0.305
% increase	0%	19.06%	26.16%+	39.75%++	37.83%++	42.50%++
2012 err@10	0.158	0.172	0.175	0.196	0.200	0.203
% increase	0%	8.54%	10.88%	23.66%	26.19%++	28.65%++
2011 ndcg@10	0.318	0.330	0.378	0.431	0.431	0.420
% increase	0%	4%	19%++	36%++	36%++	32%++
2011 err@10	0.246	0.249	0.289	0.320	0.329	0.322
% increase	0%	1%	17%	30%+	34%++	31%+

Table 4.5: α-nDCG@10 and ERR-IA@10 using Bing and combinations Bing and session-dependent data.

	LM	Bing	B+Query	B+Snip	B+Xtrk	B+alch
2011 α-ndcg@10	0.3737	0.391	0.440	0.510	0.512	0.493
% increase	0%	4.49%	17.46%++	36.27%++	36.98%++	31.85%++
2011 err-IA@10	0.3325	0.349	0.394	0.477	0.481	0.461
%increase	0%	4.97%	18.34%+	43.30%++	44.72%++	38.62%++

Analysis

Table 4.6: nDCG@10 for query expansion + CombMNZ

	Bing+ AlchDoc 2012	Bing+ XtrakDoc 2012	Bing+ SnipDoc 2012	Bing+ AlchDoc 2011	Bing+ XtrakDoc 2011	Bing+ SnipDoc 2011
no expansion	0.305	0.295	0.299	0.420	0.431	0.431
w/ expansion	**0.314**	**0.313**	0.285	**0.422**	0.429	0.418

In this section we show results for experimenting with the sources of information that provided the best results in the "Results" section.

Effects of Query Expansion Prior to Aggregation:

In this experiment, instead of using each of the top 10 snippets retrieved by a query as related queries, we create 10 queries that contain the query text expanded with the terms in the snippet. And we do so using keywords as well. The effect is insignificant: slight increase in three cases (Table 4.6 in bold font), but slight decrease in the other three cases.

Effects of the Number of Top Documents Used:

In general, increasing the number of top documents exploited from 5 to 10 causes an improvement of the results, albeit not significantly (see Table 4.7). This could be because most of the useful data is in the top 5 documents, and only little in the next 5. Further investigation would help determine the exact reasons.

Table 4.7: nDCG@10 for different cutoffs for top documents

	Snip 2012	XtrakDoc 2012	B+Alch 2012	B+AlchDoc 2012	Xtrak 2011	XtrakDoc 2011	B+Xtrak 2011	B+XtrakDoc 2011
Top 10 docs	0.274	0.282	0.305	0.314	0.391	0.411	0.431	0.429
Top 5 docs	0.255	0.273	0.300	0.313	0.338	0.419	0.421	0.420

Table 4.8: nDCG@10 for various ways of exploiting click data

	Snip 2012	XtrakDoc 2012	B+Alch 2012	B+Alch Doc 2012	Xtrak 2011	Xtrak Doc 2011	B+Xtrak 2011	B+Xtrak Doc 2011
no clicks	0.274	0.282	0.305	0.314	0.391	0.411	0.431	0.429
clicked only	0.182	0.210	0.291	0.291	0.265	0.345	0.353	0.362
boost clicked	0.280	0.284	0.305	0.314	0.391	0.416	0.425	0.432

Effects of Clicked Data:

Aggregating using data from clicked documents alone hurts the performance significantly. However when we include queries taken from clicked documents twice (instead of including them once, as per normal), we get slightly better results than when clicks are not exploited at all (Table 4.8). That is, giving more voting rights to clicked documents improves performance. Determining by how much the voting rights must be increased is left for future work.

Table 4.9: 2011 Bing+Xtrak4Me between top-10 docs and the docs from previous interactions (Best of bing+session specific)

% of overlap	# of queries	ERR impact on queries			nDCG impact on queries		
		Incr	Same	Decr	Incr	Same	Decr
0%	41	22	7	12	19	7	15
10%	25	17	4	4	17	4	4
20%	6	2	0	4	3	0	3
30%	3	2	0	1	2	0	1
40%	1	0	0	1	0	0	1

We furthered our analyses to show that we are not merely rearranging, promoting and redisplaying documents that were shown in previous interactions of a given session. For that, we looked into document overlaps: for each session, there is only little or no overlap between our top 10 documents and the documents that are part of the previous interactions of the same session. For instance, as shown in Table 4.9, out of all 76 queries of the 2011 session dataset, 41 queries returned 10 top documents that do not overlap at all with the documents returned by the commercial engine for the previous interactions. Also, 21 out of the 41 queries witnessed an increase of ERR@10 over the baseline while only 12 witnessed a decrease and 7 remained with the same result. 25 returned 1 overlapping document out of 10. Only 1 query returned the maximum of 4 documents out of 10 that overlap with previous interaction

documents. This suggests that we are not merely redisplaying the documents displayed to users in previous interactions when the session data was being collected.

Table 4.10: nDCG@10 per task type for 2012 session using session-dependent data for CombMNZ

	LM	Query	Snippet	XtraK	Alchemy
amorphous	0.2512	0.2707	0.2899	0.2659	0.2217
% change	0%	7.76%	15.41%	5.85%	-11.74%
factual	0.2215	0.2457	0.2631	0.2287	0.2526
% change	0%	10.93%	18.78%	3.25%	14.04%
intellectual	0.2129	0.2546	0.2996	0.247	0.2266
% change	0%	**19.59%**	**40.72%**	**16.02%**	6.43%
specific	0.1943	0.2318	0.2654	0.2112	0.2602
% change	0%	19.30%	36.59%	8.70%	**33.92%**

For the 2012 session dataset, the topics are categorized under different types depending on their search goal (specific or amorphous goal) and their product type (factual or intellectual target). We evaluate our results for each task type and show in table 4.10 and 4.11 how the effects of our methods differ on each task type. In general, our methods achieve their biggest improvements on the intellectual session searches with a peak of 67.5% improvement over the baseline's nDCG@10 when using the combination of XtraK4Me keyphrases and Bing's related queries (B+Xtrk). Even when using session-dependent data only, we achieve a significant peak of 40.72% increase of the nDCG@10. Intellectual searches are more difficult than factual searches, so our systems perform better on the most difficult product type sessions. While the dirichlet-smoothed Language model gets 0.2215 for factual tasks and only 0.2129 for intellectual, B+Xtrk gets 0.2804 for factual and a very high 0.3408 for intellectual tasks comparatively. This may be because our system uncovers hidden interesting keywords that were not obvious from intellectual task topic description.

However, it is noteworthy that our methods also gets significant improvements on the other task types.

In a related effort, we strive to determine how many tokens introduced by our methods are not part of the main query, but are part of the topic description. Stop words are excluded from the lists of tokens. As can be seen in Figure 4.4, Snip, Xtrak and Alch methods introduce a sizeable number of tokens that were not part of the *current-query* and would have hence been overlooked even though they are potentially important (since they are part of the topic description). In fact, Figure 4.3 shows that our method is successful at promoting documents that cover two or more aspects of a topic. Out of the total 76 sessions, 35 witness Xtrak4Me doing better than the baseline in promoting documents that cover two or more aspects, 20 witness the inverse, and 21 witness no difference.

Table 4.11: nDCG@10 per task type for 2012 session using Bing alone and combinations of Bing and session-dependent data for CombMNZ. B is short for Bing

	LM	Bing	B+Query	B+Snip	B+Xtrk	B+alch
amorphous	0.2512	0.3064	0.3152	0.3255	0.3391	0.3242
% change	0%	21.97%	25.48%	29.58%	34.99%	29.06%
factual	0.2215	0.2309	0.252	0.2804	0.2595	0.2831
% change	0%	4.24%	13.77%	26.59%	17.16%	27.81%
intellectual	0.2129	0.301	0.3029	0.3408	0.3566	0.3426
% change	0%	**41.38%**	**42.27%**	**60.08%**	**67.50%**	**60.92%**
specific	0.1943	0.2165	0.2349	0.2836	0.259	0.2888
% change	0%	11.43%	20.90%	45.96%	33.30%	48.64%

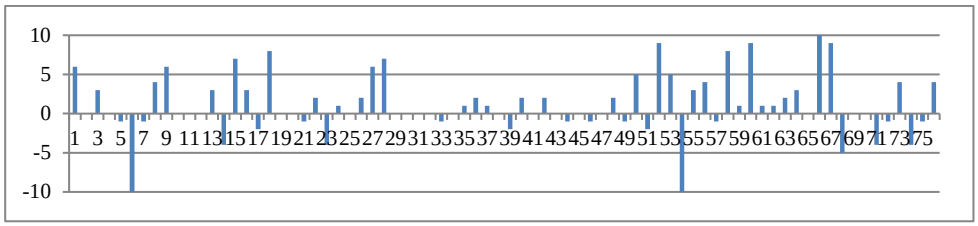


Figure 4.3: Difference between Xtrak4Me and LM in terms of the number of documents that cover 2 or more aspects (y-axis) for all sessions, on the 2011 dataset.

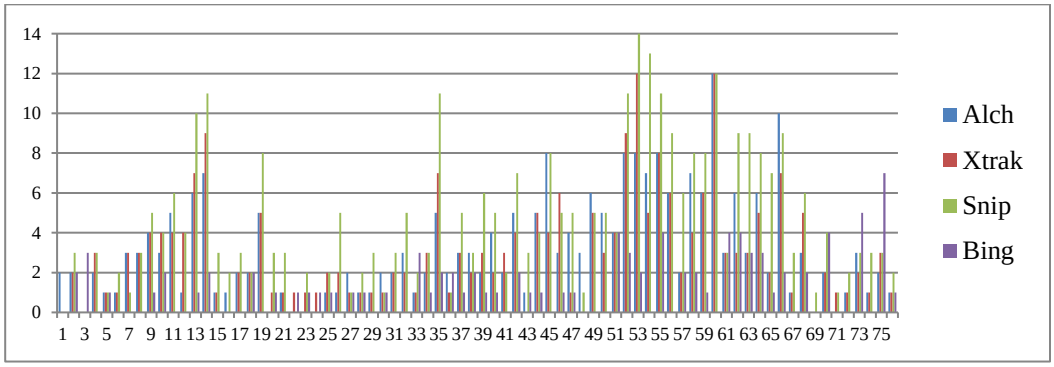


Figure 4.4: Number of tokens introduced by Alch, Xtrak, Snip and Bing that are not part of the main query, but are part of the topic description on the 2011 dataset.

4.4 Summary

We showed that using a simple data fusion method over a good set of related queries helps improve results and we show which sources are useful for collecting good related queries. We used a data fusion method that promotes documents that are likely to cover high numbers of subtopics relevant to the information need; as well as two existing data fusion methods. We found that using Bing's related queries is a good choice, but using session-dependent data is even better. Furthermore we achieve even better results by combining Bing related queries to session-dependent queries.

Chapter 5

EFFECTIVE AND ROBUST MODELS FOR SEARCH AND RETRIEVAL

Most research in Information Retrieval, such as our work in the previous chapter, has focused on improving the average effectiveness of systems. However, it is very often the case that the improved systems fare worse than the baseline on certain queries, even though the average effectiveness score is higher than the baseline's. The concept of robust ranking appears therefore to be key, when it comes to remedying those cases. In this present work, robustness refers to the ability of the ranker to reduce and mitigate poor performance on individual queries while striving to improve the overall performance as well.

In this chapter, we propose two re-ranking techniques – based on exploiting the popularity of documents with respect to a general topic – that, given a baseline or a state-of-the-art ranking, improves the average effectiveness of the ranking while improving the robustness of the ranking. Both methods merely re-rank the documents that were retrieved by the baseline that is being considered, without ever adding any new document to the set of retrieved documents. We used each of the runs submitted to TREC Web tracks 2013-14 as baseline, and empirically show that our algorithms improve the effectiveness as well as the robustness of the systems in an overwhelming number of cases, even though the systems used to produce them employ a variety of retrieval models.

We start by describing the task and the datasets used.

5.1 Retrieval task and Data

Each of the TREC Web track 2013 and 2014 datasets contains 50 queries [41, 42]. Those queries were created after perusing candidate topics from query logs from commercial search engines. Some queries were faceted (i.e. their topic has several possible subtopics), others were non-faceted with single intents, and a few others were ambiguous (i.e. queries with several intents). The task of the participants is to provide a diversified ranking of no more than 10000 documents per query for the 50 queries. Unlike in previous versions of the track that ran from 2009 to 2011 where the emphasis was on diversity ranking in addition to ad-hoc ranking, in these two versions, the emphasis was on risk-minimization and ad-hoc rankings. However the queries and relevance judgments were still suitable for diversity retrieval evaluations, and participants were also provided with diversity-based results once the competition was over. In this chapter, we use diversity measures to show how effective our method is, and we use risk-sensitive measures adopted by TREC Web track organizers to show robustness.

For this experiment, in order to show that our method can be applied to many retrieval models, we use many baseline ranked lists (RL). Specifically, we use RLs submitted by TREC Web 2013 and 2014 participants, each RL is created by a different search system. For the 2013 TREC Web track, there were a total of 60 runs, while for 2014, there were a total of 30 ad-hoc runs and 12 risk-sensitive runs. All runs were supposed to be risk-sensitive, although some participants did not specifically optimize for risk-sensitivity [41, 42].

Baselines for computing risk-sensitive measures were also made public by the track organizers, and include Indri, Terrier, Indri-with-spam-filtering. The idea behind using several baselines is to see how truly robust a system is with respect to various

baselines, and mitigate the bias that gets introduced when using only one baseline (in such a case, there would be bias towards systems that are built on top of a ranker similar to the baseline).

In summary, we used each of the runs submitted to TREC Web tracks 2013-14 as baseline, and empirically show that our algorithms improve the effectiveness as well as the robustness of the systems in an overwhelming number of cases, even though the systems used to produce them employ a variety of retrieval models.

5.2 Retrieval Model

In order to show that we can improve the ranked list for a given query by exploiting a prefetched list of documents sorted by decreasing probability of "retrievability" of a document (see below for how we estimate retrievability), we proceed as follows:

a. For obtaining pre-fetched lists of documents ranked by decreasing probability of retrievability, we propose to use the method described in section 5.2.1.
b. In order to obtain a baseline ranked-list for each given query, we use every ranked list submitted at TREC Web tracks 2013 and 2014 as baseline. That allows us to show that our re-ranking works on a wide-range of systems.
c. Then at query run-time, we propose to use one of the two algorithms proposed in section 5.2.2 for obtaining the final ranked list.

Finally, we evaluate the effectiveness of each method on each baseline using diversity measures and the robustness using risk-sensitive measures.

5.2.1 Estimating Document "Retrievability"

Given a query, we want to estimate how likely a document is to be retrieved. That is, we want to estimate its retrievability with respect to the topic. We treat this as the popularity of the document with respect to that general topic.

Suppose q is the observed query, that is, an actual user query. Let $\mathcal{Q}$ represent the space of possible queries from which q is one sample. In our experiments, we obtain the sample of possible queries by using Bing and Yahoo! Suggestions. We submit a query to each service through their APIs and we obtain a list of suggested queries.

Now let us say a document is *retrieved* if it appears in a top-k ranking of documents for some query q. In our experiments, we use k=100.

We cannot observe that space fully, but we assume that the probability that a document is retrieved for a query in that space is approximately the same as the probability that a document is retrieved for q:

$$P(ret \mid q' \in Q, D) \sim P(ret \mid q \in Q, D)$$

Then let us define the probability that a document is *retrievable* for the possible query space as:

$$P(ret \mid \mathcal{Q}, \mathrm{D}) = \sum_{q' \in \mathcal{Q}} P(ret \mid q', D) P(q' \mid \mathcal{Q}, \mathrm{D})$$

If we decide to use uniform weights for P(q' | $\mathcal{Q}$,D) and if we assume that P(ret | q',D) = 1 if document D appears in top-k ranking for q' and P(ret | q',D) = 0 otherwise, then this equation is proportional to the CombCAT data fusion method that we introduced in Chapter 4 [6], which merges retrieval results based on the total number of times a document appears. We can think of – and use – this probability of

retrieval of a document as a type of popularity score for the document with respect to the sample space.

5.2.2 Reranking

We propose two different methods for re-ranking the baseline.

5.2.2.1 Method 1

Suppose a user provides a query q to our system. Given a pre-fetched ranked list in decreasing order of P(ret | q', D) – which we name MasterList – of documents pertaining to the same topic as q, we proceed as illustrated in Algorithm 1 (Figure 5.1) and Figure 5.2.

```
MasterList = docs ranked in decreasing order of P(ret | Q, D);
MasterList= MasterList minus docs that appeared in the ranked
list of only one possible query q' in our sample space;
DocsToShuffle = {}

//record docs that must be shuffled
For each doc_i in baseline
   If  doc_i is in MasterList
      Add doc_i to DocsToShuffle;

//proceed to the actual shuffling
nextIndex = 0;
For i in baseline.size()
   If  doc(i) is in DocsToShuffle
    doc(i)= DocsToShuffle(nextIndex);
       nextIndex++;
```

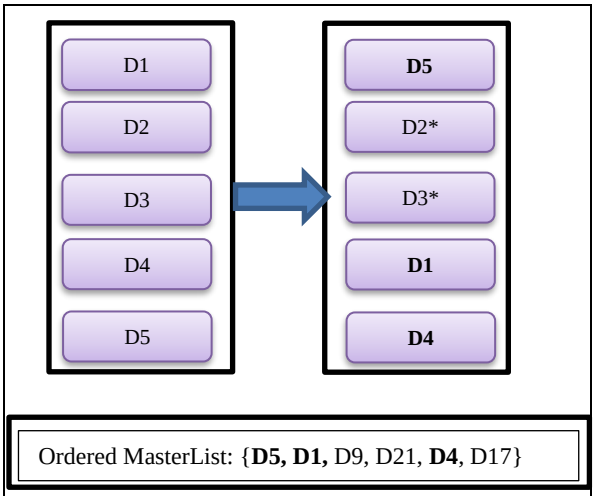
Figure 5.1: Algorithm 1

The idea is to keep, in our final ranked list, all documents that do not appear in MasterList, at the same position where they appeared in the baseline ranked list (RL). The only documents to be shuffled are the ones that:

- appeared in both the MasterList and the baseline ranked list;

- appeared in the ranked list of more than one possible query q' from the sample space $\mathcal{Q}$.

The shuffling will be done such that the documents with higher P(ret | $\mathcal{Q}$, D) – as recorded in the MasterList – will be ranked higher than the documents with lower P(ret | $\mathcal{Q}$, D) in the final ranked list. But again, the documents that are in the baseline RL but not in the MasterList, remain at their original rank.



Note: Resulting ranking is on the right. Documents marked with * retain their position since they did not appear in MasterList. The ones in bold are the intersection between MasterList and baseline ranked list.

Figure 5.2: Illustration of Algorithm 1

5.2.2.2 Method 2

Given a query q provided by the user to our system and a MasterList that contains a pre-fetched list of documents ranked in decreasing order of P(ret | q', D), we proceed as illustrated in Algorithm 2 (Figure 5.3) and Figure 5.4.

```
MasterList = docs ranked in decreasing order of P(ret | Q,D);

MasterList = MasterList minus docs that appeared in the ranked
list of only one possible query q' in our sample space;

IntersectDocs ={}

/*record docs that are both in MasterList and in baseline
ranking */
For each doc_i in baselineList
   If  doc_i is in MasterList
      Add doc_i to IntersectDocs;

/* Start by first ranking the docs from IntersectDocs in the
order the appear in (decreasing order of P(ret | q', D)*/
nextIndex = 0;
For i in IntersectDocs.size()
   doc(i)= IntersectDocs (i);

For j = 1 to  rankCutoff + IntersectDocs.size()
   If IntersectDocs does not contain baselineList(j)
      doc(j + IntersectDocs.size()) = (baselineList(j)
      doc(i) = IntersectDocs (i);
```

Figure 5.3: Algorithm 2

The essential idea in this method is to ensure that the documents with higher P(ret | $\mathcal{Q}$, D) – as recorded in the MasterList – will be ranked higher than the documents with lower P(ret | $\mathcal{Q}$, D) in the final ranked list. And, in this method – unlike in Method 1 – all the documents that are at the intersection of MasterList and the baseline RL will have precedence over all other documents in the baseline RL. That is, every document that appeared in the ranked list of more than one possible query q' from the sample space $\mathcal{Q}$, and that also appeared in the baseline RL, will be ranked before all other documents.

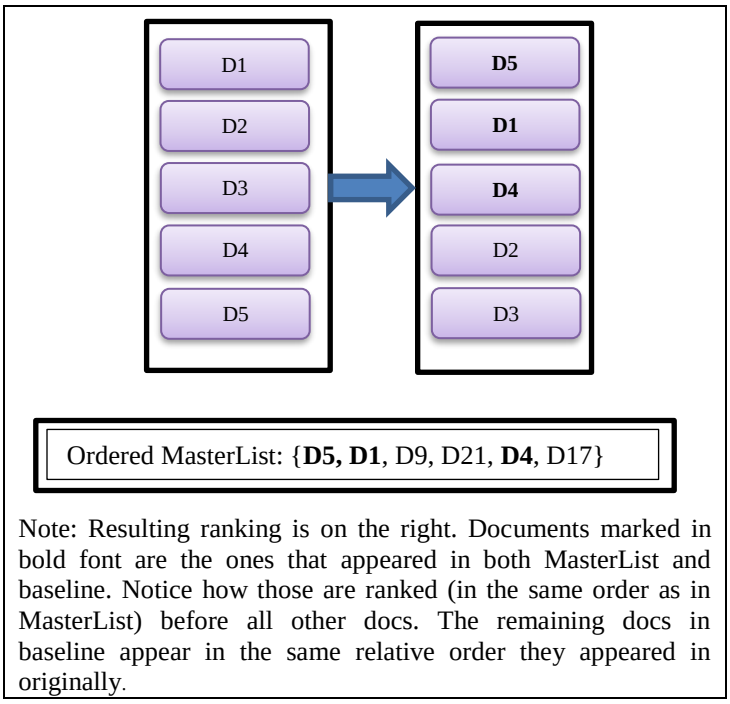


Figure 5.4: Illustration of Algorithm 2

5.3 Evaluation Measures

We use four evaluation measures (two for diversity evaluation and two for traditional non-diversity evaluation). For non-diversity measure, we opted for using the two measures adopted by TREC Web track organizers: ERR and nDCG. nDCG rewards documents with high relevance grades and discounts the gains of documents that are ranked at lower positions [53]. ERR is defined as the expected reciprocal length of time it takes the user to find a relevant document [36], and it takes into account the position of the document as well as the relevance of the documents shown above it. For diversity measures, we opted for using α-nDCG and ERR-IA. α-nDCG is an extension of nDCG that rewards novelty and diversity by penalizing redundancy

and rewarding systems for including new subtopics [39]. Similarly, ERR-IA is an extension of ERR to compute the expectation of ERR over the different intents [35].

As for measuring the robustness of the systems, we adopted the risk-sensitive measures proposed by TREC Web track organizers as well [42]. For each run, we create two new runs using the re-ranking methods Method1 and Method2 respectively. For each query of each new run, we compute the absolute difference (Δ) between the effectiveness of the new run and that of the baseline provided by the track organizers – as mentioned in 4.1, that can be either Indri or Terrier or Indri-with-spam-filtering. When the difference is positive the new run has a win over the baseline. When it is negative, it has a loss over the baseline, otherwise it is a tie.

Let $\Delta(q) = R_A(q) - R_{BASE}(q)$ be the absolute win or loss for query q with system retrieval effectiveness $R_A(q)$ relative to the baseline's effectiveness $R_{BASE}(q)$ for the same query. We define the risk-sensitive utility measure $U_{RISK}(q)$ of a system over a set of queries Q as:

$$U_{RISK}(Q) = \frac{1}{N}\left[\sum_{q \in Q_+} \Delta(q) - (\alpha + 1)\sum_{q \epsilon Q_-} \Delta(q)\right]$$

where Q_+ is the set of queries for which $\Delta(q) > 0$ and Q_- is the set of queries for which the $\Delta(q) < 0$.

It is important to note that we did not need to apply the guideline given by [46] to mitigate bias by using the mean within-topic system effectiveness as a baseline. In fact, we focus on showing that, for most rankings obtained using specific retrieval models, we can apply our method to improve the robustness of the system as well as the overall effectiveness. And we do so by comparing the risk-sensitive measure between the original ranking and the baseline to the risk-sensitive measure between the new re-ranking and the baseline.

5.4 Experiments and Results

5.4.1 Effectiveness

Table 5.1: TREC 2013 results for **Method2**. Bold font denotes positive difference between Method2 and the submitted-run, in terms of ERR-IA@20. + denotes statistical significance

runID	baselineERR-IA@20	ERR-IA@20	runID	baselineERR-IA@20	ERR-IA@20
Clustmrfaf	0.5540	**0.5701**	udemQlml1FbR	0.4620	**0.5769+**
Clustmrfbf	0.4888	**0.5266**	udemQlml1R	0.4597	**0.5706+**
cwiwt13cpe	0.4082	**0.5575+**	UDInfolabWEB1	0.4856	**0.6013+**
cwiwt13cps	0.4726	**0.5692+**	UDInfolabWEB1R	0.4856	**0.6013+**
cwiwt13kld	0.3274	**0.5525+**	UDInfolabWEB2	0.5738	**0.6013**
Dlde	0.0453	**0.2411+**	UDInfolabWEB2R	0.5738	**0.6013**
ICTNET13ADR1	0.4743	**0.5777+**	UJS13LCRAd1	0.4265	**0.5266+**
ICTNET13ADR2	0.4925	**0.5712+**	UJS13LCRAd2	0.4580	**0.5266**
ICTNET13ADR3	0.4415	**0.5784+**	UJS13Risk1	0.4435	**0.5266+**
ICTNET13RSR1	0.5185	**0.5799**	UJS13Risk2	0.4606	**0.5266**
ICTNET13RSR2	0.4847	**0.5915+**	uogTrADnLrb	0.5123	**0.5645**
ICTNET13RSR3	0.5420	**0.5777**	uogTrAIwLmb	0.5391	**0.5516**
mmrbf	0.4980	**0.5266**	uogTrAS1Lb	0.5041	**0.5399**
msr_alpha0	0.3409	**0.5449+**	uogTrAS2Lb	0.5064	**0.5378**
msr_alpha0_95_4	0.3576	**0.5624+**	uogTrBDnLaxw	0.5297	0.5276
msr_alpha1	0.3565	**0.5713+**	uogTrBDnLmxw	0.5252	**0.5276**
msr_alpha10	0.3510	**0.5607+**	ut22base	0.5066	**0.6211+**
msr_alpha5	0.3515	**0.5692+**	ut22spam	0.4368	**0.6241+**
RMITSC	0.3758	**0.4488+**	ut22xact	0.5005	**0.6211+**
RMITSC75	0.3762	**0.4466+**	UWCWeb13risk01	0.2872	**0.5767+**
RMITSCTh	0.3757	**0.5531+**	UWCWeb13risk02	0.3152	**0.5767+**
udelCombUD	0.4913	**0.5701+**	webishybrid	0.3516	**0.5720+**
udelManExp	0.5086	**0.5682**	webismixed	0.4092	**0.5798+**
udelPseudo1	0.4556	**0.5701+**	webisnaive	0.3658	**0.5803+**
udelPseudo1LM	0.3758	**0.5701+**	webiswikibased	0.3812	**0.5808+**
udelPseudo2	0.5163	**0.5701**	webiswtbaseline	0.3733	**0.5809+**
udemFbWikiR	0.4746	**0.5703+**	wistud.runA	0.4379	**0.5830+**
udemQlm1l	0.4597	**0.5706+**	wistud.runB	0.4523	**0.5642+**
udemQlm1lFb	0.4014	**0.5705+**	wistud.runC	0.3870	**0.3871**
udemQlm1lFbWiki	0.4746	**0.5703+**	wistud.runD	0.5026	**0.5731**

Effectiveness results, given by diversity measures α-nDCG@20 and ERR-IA@20, show that our approach is very promising. The results for applying Methdod2 on the 2013 runs show that, in most cases, there are large improvements. In fact, out of the 60 runs, only one of them saw a slight decrease in ERR-IA@20 using Method 2. The general trend, indeed, is that Method2 performs well on both datasets – 59 runs improved out of 60 for the 2013 dataset, 28 runs improved out of the 30 runs for 2014 dataset (ad-hoc runs category) and 9 runs improved out of the 12 runs for 2014 dataset (risk-sensitive runs category). It is worth noting that the runs from 2014 dataset that Method2 failed to improve are all from the same participating group that performed the best. This could have to do with the set of documents retrieved for their runs. Results for α-nDCG@20 have similar trends as results for ERR-IA, as shown in Table 5.2.

Table 5.2: Summary of effectiveness. This shows, for each measure for each dataset, the number of runs for which a given method performs better (on average) than another method – and vice versa. Alg1 stands for method1. Base is short for baseline

	α-nDCG@20 all 2013	ERR-IA@20 all 2013	α-nDCG@20 adhoc 2014	ERR-IA@20 adhoc 2014	α-nDCG@20 risk-runs 2014	ERR-IA@20 risk-runs 2014
# (alg1>base)	**57**	**55**	**26**	**27**	**8**	**7**
# (alg1<base)	3	5	4	3	4	5
# (alg2>base)	**58**	**59**	**27**	**28**	**9**	**9**
# (alg2<base)	2	1	3	2	3	3

Table 5.2 also shows results produced when applying Method 1. They are very close to the ones obtained using Method 2, albeit slightly lower – using ERR-IA@20,

55 results improved vs 59 for Method2 on 2013 runs, 27 vs 28 on 2014 ad-hoc runs, and 7 vs 9 on 2014 risk-sensitive runs. However, the actual numbers for Method1 effectiveness (shown in Table 5.5) are much lower than the numbers for Method2.

5.4.2 Risk analysis

Summary of risk-sensitive measures shown in Table 5.3 are evidence that our methods, overall, improve robustness as well. The effectiveness measure (R) used in the delta formula ($\Delta_q = R_A(q) - R_{BASE}(q)$) is ERR@20. We use α=5 for U_{RISK}.

However, robustness does not go up whenever effectiveness goes up. In fact, although an overwhelming number of runs witness an improvement of their robustness, this number is smaller than the number of runs for which there as improvement of the average effectiveness measures, for the 2014 runs. For the 2013 dataset, the number of improved runs based on risk-sensitive measures is very close to the number improved using average effectiveness measures: with respect to Terrier, Method1 improved robustness of 56 out of 60 runs and Method2 improved 48 out of 60. But for the TREC Web 2014 dataset, the number is much lower, especially for the runs submitted to the risk-sensitive track.

There are also stark differences in risk-sensitive measures depending on whether Terrier, Indri or Indri-with-spam-filtering is being used. The least significant improvements – as well as the most decrease in risk-sensitive measures – again are observed on the runs from the 2014 dataset. For instance, using Method1, 10 out of the 12 runs submitted for the risk-sensitive track see a decrease in risk-sensitivity utility measure with respect to Terrier, and 18 out-of-the 30 for the runs submitted to the 2014 ad-hoc track with respect to Terrier. This is not very surprising since Indri –

rather than Terrier – was used to obtain our pre-fetched list of documents sorted by popularity.

Table 5.3: Summary of Risk-sensitive results (U-ERR). This shows, for each measure for each dataset and with respect to a specific baseline, the number of runs for which a given method is more robust than another method – and vice versa. α=5

	indri all '13	indri-filt all '13	terrier all '13	indri adhoc '14	indri-filt adhoc '14	terrier adhoc '14	indri risk-runs '14	indri-filt risk-runs '14	terrier risk-runs '14
#(alg1>base)	**57**	**56**	**56**	**19**	13	12	**7**	3	2
#(alg1<base)	3	4	4	11	**17**	**18**	5	**9**	**10**
#(alg2>base)	**51**	**55**	**48**	**15**	**22**	**15**	**8**	**7**	4
#(alg2<base)	9	5	12	5	8	**15**	4	5	**8**
#(alg2>alg1)	**46**	**52**	**39**	**15**	**26**	**27**	**7**	**7**	**7**
#(alg2<alg1)	14	8	21	5	4	3	5	5	5

Table 5.4: Risk sensitive measures using indri as baseline for each run on 2013 datasets. "Alg1.UERR@20" is the risk-sensitive measure, using Method1. "Alg1-base.UERR@20" is the difference between Method1 and the submitted-run, in terms of U_{RISK}(ERR@20). α=5

runID	base. UERR@ 20	alg1. UERR@ 20	alg1-base. UERR@ 20	alg2. UERR@ 20	alg2-base. UERR@ 20	alg2-alg1. UERR@ 20
clustmrfaf	-0.0247	-0.0014	+	0.003	+	+
clustmrfbf	-0.1729	-0.1450	+	-0.180	-	-
cwiwt13cpe	-0.1102	-0.0947	+	-0.004	+	+
cwiwt13cps	-0.0655	-0.0491	+	0.003	+	+
cwiwt13kld	-0.1586	-0.0747	+	-0.008	+	+
dlde	-0.5465	-0.5444	+	-0.410	+	+
ICTNET13ADR1	-0.1166	-0.0232	+	-0.017	+	+
ICTNET13ADR2	-0.1221	-0.0184	+	-0.017	+	+
ICTNET13ADR3	-0.1790	-0.0369	+	-0.018	+	+
ICTNET13RSR1	-0.1291	-0.0431	+	-0.007	+	+

ICTNET13RSR2	-0.1116	-0.0404	+	-0.002	+	+
ICTNET13RSR3	-0.0783	0.0066	+	-0.017	+	-
mmrbf	-0.1727	-0.1441	+	-0.180	-	-
msr_alpha0	-0.1967	-0.0261	+	-0.025	+	+
msr_alpha0_95_4	-0.1781	-0.0249	+	-0.023	+	+
msr_alpha1	-0.1750	-0.0254	+	-0.023	+	+
msr_alpha10	-0.1812	-0.0261	+	-0.025	+	+
msr_alpha5	-0.1880	-0.0335	+	-0.032	+	+
RMITSC	-0.0202	-0.0118	+	-0.094	-	-
RMITSC75	-0.0213	-0.0140	+	-0.103	-	-
RMITSCTh	-0.0200	-0.0006	+	-0.018	+	-
udelCombUD	-0.1267	-0.0032	+	0.003	+	+
udelManExp	-0.1002	-0.0087	+	-0.006	+	+
udelPseudo1	-0.1933	-0.0040	+	0.003	+	+
udelPseudo1LM	-0.2632	-0.0050	+	0.003	+	+
udelPseudo2	-0.1238	-0.0380	+	0.003	+	+
udemFbWikiR	-0.0610	-0.0430	+	0.003	+	+
udemQlm1l	-0.0570	-0.0107	+	0.004	+	+
udemQlm1lFb	-0.0990	-0.0375	+	0.004	+	+
udemQlm1lFbWik i	-0.0610	-0.0430	+	0.003	+	+
udemQlml1FbR	-0.1037	-0.0223	+	-0.019	+	+
udemQlml1R	-0.0570	-0.0107	+	0.004	+	+
UDInfolabWEB1	-0.2053	-0.0722	+	-0.039	+	+
UDInfolabWEB1R	-0.2053	-0.0722	+	-0.039	+	+
UDInfolabWEB2	-0.0897	-0.0425	+	-0.039	+	+
UDInfolabWEB2R	-0.0897	-0.0425	+	-0.039	+	+
UJS13LCRAd1	-0.2446	-0.1838	+	-0.180	+	+
UJS13LCRAd2	-0.2116	-0.1603	+	-0.180	+	-
UJS13Risk1	-0.2340	-0.1638	+	-0.180	+	-
UJS13Risk2	-0.2206	-0.1528	+	-0.180	+	-
uogTrADnLrb	-0.0425	-0.0416	+	-0.030	+	+
uogTrAIwLmb	-0.0662	-0.0823	-	-0.017	+	+
uogTrAS1Lb	-0.0514	-0.0451	+	-0.042	+	+
uogTrAS2Lb	-0.0673	-0.0738	-	-0.044	+	+
uogTrBDnLaxw	-0.1671	-0.1460	+	-0.180	-	-
uogTrBDnLmxw	-0.1641	-0.1245	+	-0.180	-	-
ut22base	-0.0651	-0.0534	+	-0.080	-	-
ut22spam	-0.1842	-0.1517	+	-0.061	+	+
ut22xact	-0.0453	-0.0362	+	-0.076	-	-
UWCWEB13RIS K01	-0.2978	-0.0301	+	-0.021	+	+
UWCWEB13RIS	-0.2362	-0.0449	+	-0.021	+	+

K02						
webishybrid	-0.1240	-0.1055	+	-0.008	+	+
webismixed	-0.1069	-0.0917	+	-0.001	+	+
webisnaive	-0.1356	-0.1095	+	-0.001	+	+
webisrandom	-0.1182	-0.0867	+	0.003	+	+
webiswikibased	-0.1255	-0.1052	+	-0.001	+	+
webiswtbaseline	-0.1333	-0.1048	+	-0.001	+	+
wistud.runA	-0.1299	-0.0410	+	-0.038	+	+
wistud.runB	-0.1656	-0.0380	+	-0.036	+	+
wistud.runC	-0.2830	-0.2900	-	-0.286	-	+
wistud.runD	-0.0442	0.0232	+	-0.001	+	-

Table 5.5: TREC 2013 results for **Method1**. Bold font denotes positive difference between Method2 and the submitted-run, in terms of ERR-IA@20. + denotes statistical significance

runID	baselineERR-IA@20	ERR-IA@20	runID	baselineERR-IA@20	ERR-IA@20
clustmrfaf	0.5540	**0.5994**	udemQlml1FbR	0.4620	**0.5552**
clustmrfbf	0.4888	**0.5574**	udemQlml1R	0.4597	**0.5221**
cwiwt13cpe	0.4082	**0.4356**	UDInfolabWEB1	0.4856	**0.5943**
cwiwt13cps	0.4726	**0.5045**	UDInfolabWEB1R	0.4856	**0.5943**
cwiwt13kld	0.3274	**0.3903**	UDInfolabWEB2	0.5738	**0.6048**
dlde	0.0453	**0.0464**	UDInfolabWEB2R	0.5738	**0.6048**
ICTNET13ADR1	0.4743	**0.5432**	UJS13LCRAd1	0.4265	**0.4890**
ICTNET13ADR2	0.4925	**0.5461**	UJS13LCRAd2	0.4580	**0.5025**
ICTNET13ADR3	0.4415	**0.5541**	UJS13Risk1	0.4435	**0.5193**
ICTNET13RSR1	0.5185	**0.5513**	UJS13Risk2	0.4606	**0.5436**
ICTNET13RSR2	0.4847	**0.4998**	uogTrADnLrb	0.5123	0.4895
ICTNET13RSR3	0.5420	**0.5963**	uogTrAIwLmb	0.5391	0.5182
mmrbf	0.4980	**0.5592**	uogTrAS1Lb	0.5041	0.4968
msr_alpha0	0.3409	**0.5374**	uogTrAS2Lb	0.5064	0.4972
msr_alpha0_95_4	0.3576	**0.5568**	uogTrBDnLaxw	0.5297	**0.5425**
msr_alpha1	0.3565	**0.5537**	uogTrBDnLmxw	0.5252	**0.5532**
msr_alpha10	0.3510	**0.5598**	ut22base	0.5066	**0.5235**
msr_alpha5	0.3515	**0.5645**	ut22spam	0.4368	**0.4657**
RMITSC	0.3758	**0.4110**	ut22xact	0.5005	**0.5424**
RMITSC75	0.3762	**0.4113**	UWCWeb13risk01	0.2872	**0.5541**
RMITSCTh	0.3757	**0.4148**	UWCWeb13risk02	0.3152	**0.5490**
udelCombUD	0.4913	**0.5685**	webishybrid	0.3516	**0.3963**
udelManExp	0.5086	**0.5575**	webismixed	0.4092	**0.4362**

udelPseudo1	0.4556	**0.5705**	webisnaive	0.3658	**0.4098**
udelPseudo1LM	0.3758	**0.5680**	webiswikibased	0.3812	**0.4274**
udelPseudo2	0.5163	**0.5517**	webiswtbaseline	0.3733	**0.4094**
udemFbWikiR	0.4746	**0.5022**	wistud.runA	0.4379	**0.5707**
udemQlm1l	0.4597	**0.5221**	wistud.runB	0.4523	**0.5642**
udemQlm1lFb	0.4014	**0.4797**	wistud.runC	0.3870	0.3829
udemQlm1lFbWiki	0.4746	**0.5022**	wistud.runD	0.5026	**0.5806**

5.5 Summary

We proposed two re-ranking approaches based on exploiting document "popularity" across a topic, and show that these methods can help improve average overall effectiveness as well as robustness. Using the runs submitted to TREC Web track 2013 and 2014 as baselines, we show that, after our re-ranking, overall effectiveness gets improved in an overwhelming number of cases, and robustness gets improved in a large number of cases but fewer than for overall effectiveness.

Chapter 6

GENERAL THEORETICAL FRAMEWORK FOR RETRIEVAL AND RANKING FOR SEARCH OVER SESSIONS, AD-HOC AND DIVERSITY RANKING

As we have seen in Chapter 4, data fusion has been shown to be a simple and effective way to improve retrieval results. But the techniques we have presented are somewhat ad hoc and lacking in underlying principles that might explain their efficacy. Retrieval models explicitly derived on principles such as probability theory or linear algebra have the advantage of building on well-understood definitions and theorems, as well as providing a more general framework to build on. We thus propose a framework based partially on the well-known Probability Ranking Principle, which says that the optimal ranking of documents is achieved when sorting them in decreasing order of probability of relevance [84]. We augment it with the following guideline: optimal ranking is achieved when documents are ranked in decreasing order of probability of relevance and "*retrievability*".

We use our model to also address the problems of novelty and diversity ranking. This is motivated by the fact that ideas from data fusion have found their way into novelty and diversity ranking as well. Novelty retrieval aims at reducing redundancy in search results, while diversity retrieval aims to handle query ambiguity. Many methods for novelty and diversity retrieval attempt to identify possible intents or subtopics of a query, find documents relevant to those, and then combine them into a single ranked list. The difference from traditional data fusion methodologies is that the combination procedure attempts to account for possible redundancy.

In practice, users sometimes solve the novelty/diversity problems themselves by simply reformulating their query several times over a session of interactions. In those cases the user is the fusion algorithm, mentally keeping track of the relevant documents they've seen over the course of the session. In such cases the user is fusing results from different query expressions of the same information need from the same retrieval engine, rather than fusing results from different retrieval engines for the same query.

What if instead the system was able to use the user's historical interactions with it to anticipate the user's needs and produce more relevant documents faster? After all, the user described in the previous paragraph may have a perfect understanding of their information need, but cannot possibly understand the extent of the full corpus as well as the system can. The system could use information from the user and other similar users to generate queries to produce results that can be literally fused for the user.

Our theoretical probabilistic framework for data fusion is based on combining ranked lists from different queries that users could have entered for their information need. If we can identify a set of "possible queries" for an information need, and estimate probability distributions concerning the probability of generating those queries, the probability of retrieving certain documents for those queries, and the probability of documents being relevant to that information need, we have the potential to dramatically improve results over a baseline system given a single user query. Our framework is based on several component models that can be mixed and matched. We present several simple estimation methods for components. Using our framework, we were able to achieve significant results that are competitive with, and

in some cases outperform the state-of-the-art methods known in the literature, using TREC Web 2013 and 2014 [41], [42], NTCIR IMine 2014 [70] and TREC Session track 2013 and 2014 datasets [32], [31].

6.1 Model

Our proposed model is a probabilistic framework inspired by both the Probability Ranking Principle (PRP) of Robertson [84] and the idea of data fusion. It is based on obtaining a collection of "possible queries" for an information need, obtaining ranked lists for each of those queries, then fusing ranked results in different ways.

The traditional PRP says that the optimal ranking of documents is in decreasing order of their relevance to the user's information need. For a given user need expressed as a query q and a particular document D, the probability that D is relevant is expressed as $P(\text{rel}|q, D)$. In most classical IR settings, the query q is the only available evidence about the user's need.

Language modeling approaches estimate $P(\text{rel}|q, D)$ as the probability of sampling the query q from a language model computed from the document D and other evidence, that is, $P(\text{rel}|q, D) \approx P(q|D)$. In our framework, we suppose instead that q has been sampled from larger space of possible queries that a user might have chosen for their original information need. Let us denote this space $\mathcal{Q}$. Then we could modify the PRP to rank documents in decreasing order of $P(\text{rel}|\mathcal{Q}, D)$. If q is still the only available evidence about the user's need, then $P(\text{rel}|\mathcal{Q}, D) = P(\text{rel}|q, D)$; the PRP is unaffected. But if there is more evidence about the space of possible queries a user might have chosen, we could expand the PRP by marginalizing over that space:

$$\mathrm{P}(\mathrm{rel}|\mathcal{Q},\mathrm{D}) = \sum_{q' \in \mathcal{Q}} P(rel|q', D)P(q'|\mathcal{Q},\mathrm{D})$$

The efficacy of this model likely depends on our ability to identify the space Q and estimate probabilities of sampling queries from it. Note, however, its similarity to the data fusion scoring method CombWSUM: the probabilities P(rel|q', D) can be seen as different retrieval scores being fused, while P(q'|$\mathcal{Q}$, D) can be seen as a weight for that score. If those weights are uniform, then the formulation is similar to CombSUM.

Now let us consider the probability that a document is retrieved rather than relevant. By analogy to the PRP, let us define P(retr|q, D) as the probability that document D is retrieved given query q. Just as we can retrospectively take P(rel|q, D) to be proportional to a score assigned by a retrieval system, we can retrospectively take P(retr|q, D) to be proportional to the rank position of the document. Using the same expansion as above, we can say:

$$\mathrm{P}(\mathrm{retr}|\mathcal{Q},\mathrm{D}) = \sum_{q' \in \mathcal{Q}} P(retr|q', D)P(q'|\mathcal{Q},\mathrm{D})$$

This allows us to use as evidence documents that have been ranked by some system for a query that is not q but that appears to be related to the same information need as q.

Note here that using uniform P (q'|$\mathcal{Q}$, D) and a simple cutoff function for P(retr|q', D) (that is, P(retr|q', D) = 1 if D is ranked above k for q' and 0 otherwise), the formulation is equivalent to the data fusion method CombCAT [6] described in Chapter 4.

Either of the probabilities P (rel|$\mathcal{Q}$, D) or P (retr|$\mathcal{Q}$, D) could be used as a scoring function by which to rank documents; their similarity to proven data fusion

methods demonstrates their effectiveness. What we propose now is combining them into a single scoring function P (retr&rel|$\mathcal{Q}$, D).

This model can be decomposed in two different ways. First, assuming that relevance and retrievability are conditionally independent given the full query sample space and a document, we obtain our first model:

$$\mathrm{M1} = \mathrm{P}(\mathrm{retr\&rel}|\mathcal{Q}, \mathrm{D}) = P(retr|\mathcal{Q}, D)P(rel|\mathcal{Q}, \mathrm{D}) \qquad (1)$$

Second, assuming that relevance and retrievability are conditionally independent given a single query and a document, we obtain our second model:

$$\begin{aligned} M_2 &= \mathrm{P}(\mathrm{retr\&rel}|\mathcal{Q}, \mathrm{D}) \\ &= \sum_{q' \in \mathcal{Q}} \mathrm{P}(\mathrm{retr\&rel}|\mathrm{q}', \mathrm{D})\mathrm{P}(\mathrm{q}'|\mathcal{Q}, \mathrm{D}) \\ &= \sum_{q' \in \mathcal{Q}} P(retr|\mathrm{q}', D)P(rel|\mathrm{q}', \mathrm{D})\mathrm{P}(\mathrm{q}'|\mathcal{Q}, \mathrm{D}) \qquad (2) \end{aligned}$$

We additionally assume that q' is independent of D (that is, that the user did not have document D in mind when formulating query q', which may indeed be unrealistic), so P (q'|$\mathcal{Q}$, D) = P (q'|$\mathcal{Q}$).

Both models combine principles of the PRP and data fusion. Their effectiveness will rely on careful implementation of several components:

- the query sample space $\mathcal{Q}$
- the probability of sampling a query from that space P (q'|$\mathcal{Q}$)
- the probability of relevance P (rel|q', D)
- the probability of retrieval P (retr|q', D)

In the following section we propose possible implementations of each of these.

6.1.1 Model components

Again, our model depends on our selection of implementations of several components. In this section we describe a few easy possibilities for each.

6.1.1.1 Query sample space Q

One of the key components of this model is the space of possible queries for an information need. This is not information that is typically readily available. It may be possible to identify related queries in a large query log, but even that will be biased to how users tend to interact with that search engine (for example, large web search engines like Google would be biased towards very short keyword queries, while a e-discovery engine might be biased towards much longer, structured Boolean queries). Thus we must approximate such a space in some way.

We consider two primary sources of information: user search history over a session, and query suggestions provided by web search engines. For the former, when the last query a user inputs in a session is q, we look back over the history of the session and assemble Q from one of the following sources:

- Previous queries in the session. These could be seen as a sample of queries by the same user from the possible space.
- Titles of documents ranked for previous queries in the session. These could be seen as approximations of verbose queries, and could potentially identify new relevant documents that would not be found without longer queries.
- Snippets of documents ranked for previous queries in the session. Again, these could be seen as very verbose queries.

The TREC Session track provides data for these; we discuss this more in Section 6.2.1 below.

We can also obtain query suggestions from external sources (e.g. via commercial query generation services that provide an API through which we submit a query and obtain a list of query suggestions for that query). These give us "indirect" access to a large query log, assuming a query log was used to generate them. And similar to using snippets from a search session, we can also use snippets generated by a commercial search engine for the top 10 documents of a query. Specifically, we investigate the following:

- Query suggestions provided by the Bing API.
- Snippets of the top-10 documents retrieved from the Yahoo! BOSS API.

We treat these as "black boxes" for generating a query sample space, giving us something like an upper bound on the performance that could be expected.

6.1.1.2 Query sample probability P (q'|$\mathcal{Q}$)

Once we have a query space $\mathcal{Q}$, we need a way to estimate the probability of sampling a particular query q' from that space. These probabilities become weights on the scores of documents retrieved for that query, or weights on whether that document was retrieved or not and at what rank.

For this work, we investigate a few simple heuristics:

- Uniform probabilities over unique queries. Each unique query has the same probability of being sampled.
- Uniform probabilities over all queries. Each non-unique query has the same probability of being sampled. This means that a query that appears twice in $\mathcal{Q}$ is twice as likely to be sampled as one that appeared only once.
- Proportional to the similarity between q' and the space $\mathcal{Q}$.
- Proportional to position in query suggestion rankings.

- The APIs we use to generate query suggestions provide them as a ranking, so we can weight them accordingly. For instance, we may assign twice as much sampling probability to query suggestions in the top half of the ranking as in the bottom half.
- Proportional to position in the session history. Queries that appear more recently may be given more weight than queries that appear further back. This can apply to titles and snippets taken from the session history as well.

The latter two are implemented by binning queries by position (either rank position or position in time). The similarity is implemented using cosine similarity. We provide more detail in Section 6.2.4 below.

6.1.1.3 Probability of relevance P(rel|q', D)

The probability of relevance is the score of the document for the sampled query q'. In general it can be any retrieval function. We have used the language model score computed by Indri, since the index of web pages that is available to us was built in Indri. Indri computes its retrieval score with a Dirichlet-smoothed language model as:

$$score(q', D) = \sum_{t \in q'} \log \frac{tf_{t,D} + \mu \frac{ctf_t}{|C|}}{|D| + \mu}$$

Here $tf_{t,D}$ is the frequency of term t in document D, ctf_t is the frequency of term t in the entire collection, |D| is the length of document D in terms, |C| is the length of the entire collection (the sum of the lengths of all documents in the collection), and is a smoothing parameter to guarantee no term has zero probability.

6.1.1.4 Probability of retrieval P(retr|q', D)

The probability of retrieval, in our work, is essentially proportional to whether a document has been ranked by a system for the query q' and at what position. We consider two possible methods to estimate this probability:

- A simple binary indicator function $I(\text{rank}_D \leq k)$. As mentioned above, we can decide that $P(\text{retr}|q', D) = 1$ if document D appears in the top-k ranking for q' and $P(\text{retr}|q', D) = 0$ otherwise.

Discounting by rank. We use a simple linear discount by which the top-ranked document is assign probability proportional to 1, then each subsequent document is discounted by an additional 1/k.

6.1.2 Connection to data fusion methods

We are proposing the models above as a framework from which new retrieval models can be produced. In this section we demonstrate how well-known fusion algorithms emerge from our framework.

One of the primary differences between our method and typical fusion methods is that we are issuing different queries to a single retrieval system, whereas fusion methods traditionally issue the same query to different retrieval systems.

Thus to connect to existing methods, we rewrite our frame-work so that it sums over different rankings rather than different queries. The modified version of M_2 (Eq. 2) looks like this:

$$\begin{aligned} M2 &= P(\text{retr\&rel}|\boldsymbol{\mathcal{R}}, D) \\ &= \sum_{R \in \boldsymbol{\mathcal{R}}} P(\text{retr\&rel}|R, D) P(R|\boldsymbol{\mathcal{R}}, D) \\ &= \sum_{R \in \boldsymbol{\mathcal{R}}} P(retr|R, D) P(rel|R, D) P(R|\boldsymbol{\mathcal{R}}, D) \end{aligned}$$

Note that we have essentially replaced q' and $\mathcal{Q}$, a query and a set of possible queries, with R and $\boldsymbol{\mathcal{R}}$, a ranking and a set of possible rankings, respectively. The theoretical frame-work makes no substantive distinction between a query that produces a ranked list and a ranked list produced from a query, which means that different queries and different retrieval systems can be treated interchangeably. The only caveat is that the different queries should be related to the same information need, at least as closely as two different retrieval systems retrieved similar documents for the same query.

6.1.2.1 CombSUM

CombSUM simply sums the scores a document received from different retrieval algorithms, then ranks documents in decreasing order of score. Since different retrieval algorithms can produce scores on very different scales, typically some form of standardization is applied to scores before summing them.

In our model, CombSUM emerges when $P(R|\boldsymbol{\mathcal{R}}, D) \propto 1$ (that is, all rankings are weighted equally), $P(\mathrm{retr}|R, D) \propto 1$ for all documents D ranked above some cutoff k in ranking R, and $P(\mathrm{rel}|R, \mathcal{Q}, D)$ is the normalized score given by the system the ranking R corresponds to. Then:

$$P(retr\&rel|\boldsymbol{\mathcal{R}}, D) = \sum_{R \in \boldsymbol{\mathcal{R}}} score(R, D)$$
$$= CombSUM(\boldsymbol{\mathcal{R}}, D)$$

6.1.2.2 CombWSUM

CombWSUM is very similar to CombSUM except that the scores are weighted, typically by some estimate of the average effectiveness of the system that

produced the score. In our model, CombWSUM is obtained by using a non-uniform probability for P (R|$\boldsymbol{\mathcal{R}}$, D). Then, labeling that probability w_R, we have:

$$P(retr\&rel|\boldsymbol{\mathcal{R}}, D) = \sum_{R \in \boldsymbol{\mathcal{R}}} w_R \cdot score(R, D)$$
$$= CombWSUM(\boldsymbol{\mathcal{R}}, D)$$

6.1.2.3 CombMNZ

CombMNZ also weights scores, but instead of weighting each individual system/document score by a system weight, it weights the sum of document scores across systems by the number of systems that gave that document a "non-zero" score (technically, the number of systems that ranked the document in the top-k retrieved). Let us refer to the number of non-zero scores as MNZ, and the sum of document scores as CombSUM as above. Then, using the first model formulation Eq. 1 above:

$$P(retr\&rel|\boldsymbol{\mathcal{R}}, D) = P(retr|\boldsymbol{\mathcal{R}}, D)P(rel|\boldsymbol{\mathcal{R}}, D)$$
$$= \sum_{R \in \boldsymbol{\mathcal{R}}} I(rank_D(R) \leq k) \cdot \sum_{R \in \boldsymbol{\mathcal{R}}} score(R, D)$$
$$= MNZ \cdot CombSUM(\boldsymbol{\mathcal{R}}, D)$$
$$= CombMNZ(\boldsymbol{\mathcal{R}}, D)$$

6.2 Experiments and Results

We performed experiments on five different datasets: the TREC 2013 and 2014 Session tracks datasets were used to further confirm the strength of our data fusion methods for search over sessions; the TREC 2013 and 2014 Web tracks, and the NTCIR 2014 iMine track were used to show the strong performance of our model on ad-hoc and diversity rankings. All five use the full ClueWeb12 corpus, though the

iMine track only used the ClueWeb12-B13 subset of it. We indexed the collection using Indri, an open-source package for indexing and retrieval that implements the inference network model [94].

6.2.1 Data

TREC Web tracks: Each of the 2013 and 2014 TREC Web track datasets contain 50 queries [41], [42] created after perusing candidate topics from query logs from commercial search engines. Some topics were faceted (with several possible subtopics), others were non-faceted with single intents, and a few others were ambiguous (queries with several intents). The task of TREC participants is to provide a diversified ranking of no more than 10000 documents per query for the 50 queries. Relevance judgments are provided at the level of topic as well as subtopic for computing diversity evaluation measures (see below).

TREC Session tracks: Like the Web track datasets, the Session track datasets contain queries. However, those queries are given in the context of user sessions consisting one or more interactions with a search engine [32], [31]. Each interaction includes a user-submitted query related to the topic, a ranked list of results from a search engine for that query, user clicks on those results, and the time spent by the user reading the clicked document. Finally, each session ends with a "current query", a query with no search engine results for which participants will provide ranked results, possibly using the session history to do so. The 2013 data includes 87 sessions, while the 2014 data include 1021 sessions, of which 100 were pooled for judging. The task is to leverage session history to rank documents for the last query (current query) in the session using the entire ClueWeb12 corpus for retrieval.

NTCIR IMine track: IMine is also a diversity-focused task. We use the English set, which consists of 50 queries for which the organizers provided query suggestions from Bing, Google and Yahoo as well as query "dimensions" generated using the method proposed by Dou et al. [47]. The task is to provide a diversified ranking of no more than 100 documents per query for the 50 queries, covering as many intents as possible [70].

6.2.2 Evaluation measures

Our primary evaluation measure is nDCG, the official measure of the TREC Session track and one of the measures for the TREC Web track. For Web track results, we also report α-nDCG, a modification of nDCG for measuring diversity of subtopics in ranked results. This measure works by penalizing redundancy in documents that appear in a ranked list [39].

6.2.3 Baselines

When we report a baseline, it is chosen from among the following: a standard Dirichlet-smoothed language model as implemented in Indri [94] and a set of runs obtained using Lavrenko & Croft's relevance models, varying the number of documents used for feedback. We choose the best-performing system from this set as the baseline.

6.2.4 System Implementations

As discussed above, we can produce different system implementations from the framework by picking and choosing among different component methods. Given the large space produced just by the possibilities we have mentioned (which are only a

subset of all possible choices), we cannot experiment and report on every combination. Here we list select combinations that we experimented with.

For the "possible queries" $\mathcal{Q}$, we mainly considered previous queries in a user session and titles of documents ranked for those queries (for the TREC Session track data) and query suggestions and snippets provided by two major search engines (for the TREC Session track data and the TREC Web track data). The NTCIR iMine track provided query suggestions from major search engines as well.

For the query sample probability $P(q'|\mathcal{Q}, D)$, we used all five of the methods described in Section 6.1.1.2 above. Cosine similarity is computed between a query q' and the full set of terms in $\mathcal{Q}$. However, because for many sources of $\mathcal{Q}$ the queries are quite short, we found that cosine similarity is not very useful in those cases. It is only useful when the queries in the set are relatively long, so we only report it in those cases.

To weight queries based on their distance in time, we use binning. We first decide on a number of bins m. Then, from a sequence of n queries making up a session, the most recent n/m are placed in the first bin, the next most recent n/m are placed in the second bin, and so on. The first bin is then given weight double that of the second bin, which receives weight double that of the third bin, and so on.

For the probability of relevance $P(\mathrm{rel}|q', D)$, we only use Dirichlet-smoothed language models. This is because our index of ClueWeb12 was built using indri, and that is the default model indri uses.

For the probability of retrieval $P(\mathrm{retr}|q', D)$, we use the rank-cutoff indicator function $I(\mathrm{rank}_D \leq k)$ and the linear discounting function described in Section 6.1.1.4.

6.3 Results

We present results organized by dataset below. Our primary results are those for the TREC Session track, since that is the dataset that provides access to session history data and thus the most direct additional information about the information need. We present results for the TREC Web track to show how our method can be adapted when session history is not directly available and to show results for novelty and diversity search. We present results for the NTCIR iMine track because that track provided suggestions from a commercial search engine (which gives precedent for our use of those queries). For each dataset, we break discussion of results out by model component.

6.3.1 Results for TREC Sessions

Table 6.1 summarizes all results for TREC Session track data. The baseline (line no. 1) is a simple Dirichlet-smoothed language model using only the query, ignoring session history and any other sources of possible information about the session or topic, though possibly using pseudo-relevance feedback as described above. The final line (no. 20) in the table is the reported best-performing system among all TREC submissions, taken from the respective TREC overview papers [32], [31]. The best TREC submission was 32% or 45% better than our baseline for 2013 and 2014 respectively, but we improve on our own weak baseline by up to 40% or 36% (respectively; both at line no. 17) for the "realistic" case of fusing results based on the user's history in the session.

6.3.1.1 Choice of possible queries $\mathcal{Q}$

Results are grouped by the selection of a source for the set of possible queries $\mathcal{Q}$. It is clear from 1 that this set has the biggest effect on performance, though how

specifically it affects performance depends on the data: for the 2013 Session track data, using document titles as a source of possible queries (lines 8 – 14) tends to outperform the use of previous queries in the session (lines 2 – 7), while for 2014, using document titles performs much worse than the use of previous queries. In both cases, however, there is a clear difference between the two sets, with another tier of results achieved by drawing from a commercial search engine (the query suggestions on lines 18 and 19, and engine snippets on lines 15 – 17); in that tier, snippets clearly outperform query suggestions.

The differences in the effectiveness of previous queries and document titles between 2013 and 2014 may be explained by differences in the engine that generated search results, differences in the user population, and differences in the amount of data. On the first point, the 2014 engine would some-times generate ranked lists that were much worse than the 2013 engine. When these poor results are used to generate new results for data fusion, it is not surprising that the end result would be worse. On the second and third points, the 2014 engine used Amazon Mechanical Turk to generate many more sessions and queries than were available in the 2013 data; it may be that the greater amount of data leads to greater effectiveness when using previous queries alone.

6.3.1.2 Choice of P(retr|q', D)

The next largest effect on performance is from the selection of "retrieval probability" P(retr|q', D). Using the rank-based linear discount function (odd-numbered lines except lines 1 and 15) outperforms the simple rank-cutoff binary function in every case that they can be directly compared. This suggests that weighting

documents by rank position *in addition* to retrieval score can improve data fusion in general.

6.3.1.3 Choice of P (q'|$\mathcal{Q}$, D)

Finally, the method for weighting queries sampled from the space has a smaller, yet still clear in some cases, effect on effectiveness. In particular, using a uniform distribution over the unique queries (lines 4 & 5, 10 & 11) outperforms a uniform distribution over all queries (lines 2 & 3, 8 & 9), meaning that giving more weight to duplicates in the set typically results in worse performance.

For the cosine similarity and binning methods, we have few cases where we can directly compare effectiveness. Cosine similarity seems to be most useful when applied to snippets from a commercial search engine, where they unequivocally improve effectiveness over the baseline (by 28% or more at line no. 16). When applied to document titles, using cosine similarity to estimate query sampling probability does not provide any clear benefit (line 14).

For the binning method, we report the maximum effectiveness over number of bins. While this may be "cheating" (in the sense that we are optimizing on the testing data), we note that the binning methods nearly always give worse effectiveness than the uniform probability methods using the same retrieval probability. Thus it seems that binning based on session history is not worthwhile.

Table 6.1: TREC Session track results for different combinations of model components. * indicates a statistically significant difference over the baseline by a paired t-test at the 0.05 level. The biggest improvement for each dataset is bolded.

no.	sample space $\mathcal{Q}$	P ($q' \mid \mathcal{Q}$, D)	P (retr$\mid q'$, D)	2013 Session nDCG@1 0	%Δ	2014 Session nDCG@1 0	%Δ
1	baseline	–	–	0.1474	–	0.1783	–
2	previous queries	Uniform	I($\text{rank}_D \leq k$) linear	0.1425	-3%	0.1849	4%
3	previous queries	Uniform	discount	0.1574	7%	0.1999	12%
4	previous queries	Unique	I($\text{rank}_D \leq k$) linear	0.1457	-1%	0.1935	9%
5	previous queries	Unique	discount	0.1574	7%	0.2057	15%
6	previous queries	bins by time	I($\text{rank}_D \leq k$) linear	0.1463	-1%	0.1874	5%
7	previous queries	bins by time	discount	0.1648	12%	0.1920	8%
8	document titles	Uniform	I($\text{rank}_D \leq k$) linear	0.1570	7%	0.1521	-15%
9	document titles	Uniform	discount	0.1609	9%	0.1587	-11%
10	document titles	Unique	I($\text{rank}_D \leq k$) linear	0.1599	8%	0.1603	-10%
11	document titles	Unique	discount	0.1635	11%	0.1699	-5%
12	document titles	bins by time	I($\text{rank}_D \leq k$) linear	0.1487	1%	0.1424	-20%
13	document titles	bins by time	discount	0.1562	6%	0.1484	-17%
14	document titles	cosine sim	I($\text{rank}_D \leq k$)	0.1538	4%	0.1501	-16%
15	external snippets	Uniform	I($\text{rank}_D \leq k$)	0.1710	16%	0.2001	12%
16	external snippets	Cosine	I($\text{rank}_D \leq k$)	0.2000	*36%	0.2277	*28%
17	external snippets	Uniform	linear discount	**0.2063**	***40%**	**0.2427**	***36%**
18	suggested queries	Uniform	I($\text{rank}_D \leq k$)	0.1589	8%	0.2261	*27%
19	suggested queries	Uniform	linear discount	0.1595	8%	0.2059	15%
20	TREC best	–	–	0.1952	*32%	0.2580	*45%

6.3.1.4 Additional Analysis: Effects on Task Types

The TREC Session track topics can be described by whether they are "factual", "intellectual", "specific", or "amorphous", referring to specific goals and products of the topic [31]. We investigated performance with different choices of Q when breaking topics out by these categories. Table 6.2 shows the results. Our methods seem to do the best job at improving results for "amorphous" and "factual" types, which represent, respectively, exploratory and fact-finding information needs. Moreover, they improve those types in both Session track datasets, suggesting that the improvement is independent of other differences between the datasets.

Table 6.2: TREC Session track results broken out by goal and product types.

	2013 Session				2014 Session			
Q	factual	intellectual	specific	amorphous	factual	intellectual	specific	amorphous
Baseline	0.1402	0.1632	0.1608	0.1141	0.1598	0.1958	0.1773	0.1797
prev. q + titles	0.1912	0.1769	0.1769	0.1954	0.1719	0.1772	0.1537	0.1965
prev. q + snips	0.2739	0.1813	0.1541	0.2584	0.1307	0.1898	0.1362	0.1876
titles + snips	0.2069	0.1541	0.1750	0.2021	0.1689	0.1604	0.1369	0.1932
All three	0.2119	0.1750	0.1750	0.2135	0.1762	0.1688	0.1427	0.2032

6.3.1.5 Additional Analysis: More about query sample space $\mathcal{Q}$

Table 6.3: TREC Session track results for different sources of possible queries $\mathcal{Q}$. * indicates a statistically significant difference over the baseline by a paired t-test at the 0.05 level. The biggest improvement for each dataset is bolded.

no.	sample space Q	2013 Session nDCG@10	%Δ	2014 Session nDCG@10	%Δ
1	Baseline	0.1474	–	0.1783	–
2	previous queries	0.1425	-3%	0.1849	4%
3	document titles	0.1570	7%	0.1521	-15%
4	Snippets	0.1376	-7%	0.1070	-40%
5	previous queries^z	0.1740	18%	0.2051	15%
6	document titles^z	0.1806	23%	0.1583	-11%
7	snippets^z	0.1803	22%	0.1347	-25%
8	prev. q + titles	0.1657	12%	0.1679	-6%
9	prev. q + snips	0.1917	30%	0.1476	-17%
10	titles + snips	0.1706	16%	0.1400	-21%
11	All	0.1847	25%	0.1600	-10%
12	prev. q + titles^z	0.1867	27%	0.1747	-2%
13	prev. q + snips^z	**0.2452**	***66%**	0.1614	-10%
14	titles + snips^z	0.1905	29%	0.1645	-8%
15	all^z	0.2004	*36%	0.1723	-3%
16	rec'ed queries	0.1589	8%	0.2261	*27%
17	external snips	0.1710	16%	0.2001	12%
18	rec. q + ext. snips	0.1920	*30%	0.2664	*49%
19	rec'ed queries^z	0.1556	6%	0.2705	*52%
20	external snips^z	0.2036	*38%	0.2505	*40%
21	rec. q + ext. snips^z	0.2256	*53%	**0.3169**	***78%**
22	TREC best	0.1952	32%	0.2580	45%

Here we discuss more findings about the choice of query sample space $\mathcal{Q}$. Notably, we show the large increase in performance when we leverage search history data from other sessions pertaining to the same topic, in addition to the user's session (all items in Table 6.3 that end with ^{z} leverage all sessions on the same topic, including

other users' sessions). We also show empirically that the performance generally increases when we combine several sources of information. Since we are only investigating the choices of query sample space here, we need to keep the other model components fixed. So we simply use the CombCAT implementation here while varying only Q. The details are below.

Lines 2–4 show performance using, respectively, previous queries in the user's session, the titles of documents retrieved for those queries, and the snippets displayed for those documents. Note that these three sets are not building on one another; the snippet set does not include titles and vice versa for instance. The three lines (i.e. 5–7) that follow these previous three lines show performance when using the same three items, with the big difference that in this case we use all sessions on the same topic, essentially vastly expanding the amount of data available for fusion. In fact, all items in Table 6.3 that end with ^{z} use all sessions on the same topic. Of course, this requires the ability to identify topically-related sessions, which is not always an easy problem—in the Session track data both session ID and topic ID are clearly marked, but this is not the case in real log data. However this helps determine whether access to a large amount of high-quality data about terms that users might think to use in queries can give gains in fused results. Here we see that using previous queries as a source of possible queries works well for both TREC 2013 and 2014 Session tracks (18% and 15% improvements respectively). Using document titles and snippets works well for 2013 (23% and 22% improvements) but not for 2014 (-11% and -25%). Note that using document titles and snippets for TREC Session 2011 and 2012 worked well as well, (see section 4.3.3). One possible reason for this is that the 2014 titles and snippets were of a lower quality than those for 2013 as a consequence of how the 2014

data was acquired [31]. We also note that none of these results are statistically significant, though the differences are large enough and consistent enough that we believe they reflect a real effect on performance.

Lines 8–11 show performance when taking the union of sets from the previous three: “prev. q + titles” creates Q using both previous queries in the session as well as titles of documents seen by the user for those queries; “prev. q + snips” creates it from queries and snippets; and so on. The “all” set unions all three sets. The next four lines (12–15) show performance when using the same four items, except that in this case we use all sessions on the same topic. Here we see that combining sources unambiguously improves results for 2013, while the performance penalty from using the poor-quality 2014 titles and snippets negates the gain from using previous queries.

Lines (16–18) use data from sources external to the session, specifically snippets and suggested queries from commercial search engines. For these, we submitted the last query in the session to Bing and Yahoo! BOSS and obtained query suggestions from the former and document snippets from the latter; these then become queries to our baseline LM system to produce ranked lists for CombCAT. Here we see that each source on its own gives good improvement (particularly for 2014, which significantly benefits from the Bing query suggestions), and when both are combined they give a substantial and significant performance boost in both datasets of 30% and 49% respectively.

The next three lines (19–21) show performance using the same three items used in the previous case, except that in this case we use all sessions on the same topic. The last line in this group again uses snippets and suggested queries, but this time takes input queries from all sessions on the same topic, essentially vastly

expanding the amount of data available for fusion. Here we see huge performance boosts: 53% and 78% on 2013 and 2014 data respectively. Of course, this requires the ability to identify topically-related sessions, which is not always an easy problem—in the Session track data both session ID and topic ID are clearly marked, but this is not the case in real log data. Nevertheless, this result suggests that access to a large amount of high-quality data about terms that users might think to use in queries can give huge gains in fused results.

Finally we compare our results to the best-performing TREC submission. The best TREC submission was 32% or 45% better than our baseline for 2013 and 2014 respectively, but we improve on our own weak baseline by up to 66% or 49% for the "realistic" case of fusing results based on the user's history in the session, or up to 78% for the less-realistic case of fusing results based on all users' sessions for the same information need.

6.3.2 Results for TREC Web

Results from the previous section demonstrate that substantial effectiveness gains are possible when using information from the user's recent search history. In many retrieval contexts there is no such history; for example, the TREC Web track provides only a single query with no session history. Can we obtain similar effectiveness improvements with that limitation?

Table 6.4 summarizes all results when using Bing query suggestions and Yahoo! BOSS snippets. (Again, these were submitted as queries to our own LM index of ClueWeb12 to provide the input to our fusion methods.) Again we see substantial and consistent improvements over our baseline, with our best performance comparable

to the best automatic TREC submissions despite the fact that our entire retrieval process is based on the weak LM baseline reported in the first line of the table.

We report both nDCG and the diversity measure α-nDCG to demonstrate the effectiveness of our methods for diversity retrieval. As Table 6.4 shows, we consistently obtain statistically significant gains for diversity as well as ad hoc retrieval.

6.3.2.1 Choice of possible queries $\mathcal{Q}$

Again, since we have no session history to draw from, we rely on query suggestions and snippets provided by external search engines. Both prove to be excellent sources for the set of possible queries, increasing effectiveness over our baseline for both 2013 and 2014 (with the 2013 results substantial and statistically significant). When used together (by taking the union of the two sets), effectiveness increases by up to 50% for the 2013 data, and 22% for the 2014 track.

We do not see as clear a difference between queries and snippets as we did between queries and titles in the Session track experiments above, though there is a clear gain from combining them.

Table 6.4: TREC Web track results for different combinations of model components. * indicates a statistically significant difference over the baseline by a paired t-test at the 0.05 level. The biggest improvement in each column is bolded. For the best reported TREC result, we report the highest value of the measure across all submissions; the system with the highest nDCG is not necessarily the same as the system with the highest α-nDCG.

			2013 Web			
$\mathcal{Q}$	P ($q' \mid \mathcal{Q}$; D)	P (retr$\mid q'$, D)	nDCG@20	%Δ	α-nDCG	%Δ
baseline	–	–	0.1863	–	0.4664	–
suggested queries	uniform	I($\text{rank}_D \leq k$)	0.2298	*23%	0.5548	*19%

		linear				
suggested queries	uniform	discount	0.2421	*30%	0.5695	22%
suggested queries	bins by rank	$I(rank_D \leq k)$	0.2235	20%	0.5447	17%
external snips	uniform	$I(rank_D \leq k)$	0.2443	*31%	0.5797	*24%
queries+snippets	uniform	$I(rank_D \leq k)$	0.2758	*48%	**0.6359**	***36%**
		linear				
queries+snippets	uniform	discount	0.2815	*51%	0.6331	*36%
TREC best	–	–	**0.3100**	**66%**	0.6280	35%
			2014 Web			
$\mathcal{Q}$	$P(q'\|\mathcal{Q}, D)$	$P(retr\|q', D)$	nDCG@20	%Δ	α-nDCG	%Δ
baseline	–	–	0.2562	–	0.5744	–
suggested queries	uniform	$I(rank_D \leq k)$	0.2558	0%	0.6645	*16%
		linear				
suggested queries	uniform	discount	0.2735	7%	0.6573	14%
suggested queries	bins by rank	$I(rank_D \leq k)$	0.2782	9%	0.6847	19%
external snips	uniform	$I(rank_D \leq k)$	0.2657	4%	0.6268	9%
queries+snippets	uniform	$I(rank_D \leq k)$	0.2997	*17%	0.6677	*16%
		linear				
queries+snippets	uniform	discount	**0.3134**	***22%**	0.6788	18%
TREC best	–	–	0.2610	2%	**0.6940**	**21%**

6.3.2.2 Choice of P (retr|q', D)

For the Web track we were able to use the same methods as we did for the Session track. Again we see that using a linear discount function always improves over a simple rank cutoff in cases where they are directly comparable.

6.3.2.3 Choice of P (q'|$\mathcal{Q}$, D)

In most cases we were only able to test the uniform probability over the set. For the case of suggested queries, we could also try binning them by their rank position in the list. This did turn out to have a positive effect: when the ranked list is split in half, with the top half receiving more weight than the bottom half, results improve over using no binning. However, the difference in effectiveness from the uniform distribution is negligible.

Note that using a uniform distribution over unique elements in the set does not have any effect for the Web track, since our suggested queries and external snippets never have any duplicates that could be removed.

6.3.3 Results for iMine 2014

The main difference between the iMine data and the other two sets is that the iMine organizers provided ready-made query suggestions that we can use in our method. We did not have much opportunity to explore other sources of queries or ways of combining them. Table 6.5 shows results, comparing the linear discount to the rank-cutoff function, again showing that linear discounting provides a benefit (though in this case very small and not significant).

Table 6.5: NTCIR iMine 2014 results using the query suggestions provided by the organizers of the task.

$\mathcal{Q}$	$P(q' \mid \mathcal{Q}, D)$	$P(retr \mid q', D)$	α -nDCG
query suggestions	uniform	$I(rank_D \leq k)$	0.6962
query suggestions	uniform	linear discount	0.6968

6.4 Summary

In this chapter, we propose a probabilistic data fusion framework, PDF, which makes a small amendment to the probability ranking function by suggesting to rank documents in decreasing order of their probability of relevance and retrieval by systems, as opposed to relevance only. We propose a way to estimate the probability that documents are retrieved by exploiting various sources of sample possible queries. We proceed to show the impacts of different choices for several model components by

implementing different instances of our model and empirically show that, when used with very rich sources of sample possible queries, they are at least on-par with the best reported systems for different search scenarios including ad-hoc search, diversity search and search over sessions. Specifically, we show them to be at least competitive with the best reported systems for TREC Session track 2013 and 2014, TREC Web track 2013 and 2014 as well as NTCIR IMine 2014 dataset.

Chapter 7

SIMULATION OF SEARCH INTERACTIONS

We have shown in Chapters 4 and 6 that users' search history can be leveraged to improve current search results. However sometimes we have little to no search history available. In such cases, it would be helpful to obtain data similar to search history data. One way of doing this is by simulating previous search interactions. Thus, in this chapter, we consider and address the problem of generating data similar to search history data in the absence of actual search history by simulating previous search interactions. In the present study, we focus on generating simulated "related queries" that can serve as an additional source of information about the current search [6]. For the sake of simplicity in our initial model, we intentionally leave out other possible exploitable resources such as clicks and dwell times for future work. We hypothesize that users reformulate their queries by leveraging some of the terms and keyphrases they find in ranked documents during their search [6], and hence we propose simple models for generating such related queries. Our study is thus focused on generating "related queries" by leveraging the most significant key-phrases from documents in our simulated interactions.

More specifically, our problem formulation is as follows: suppose we have a real user who provides one single query and nothing else. Can we generate data that can be used as substitute for real users' search history in the absence of the latter, and that leads to results similar to the ones we obtain when we leverage real users' search history?

Our contributions consist in addressing the following: Can we improve search effectiveness by leveraging simulated queries, and how does such a method compare to leveraging real search history? What are the effects of concatenating the simulated queries with the original user query and/or aggregating the resulting rankings with the ranking of the user's original query? And should we explore deeper layers in our model?

7.1 Methodology

7.1.1 A simple model for generating search history data

Our task is to generate simulated search history data that can be utilized as a substitute for real search history. Our assumption is that a user's next query reformulations are inspired and informed by the information she gets from reading the top-ranked documents from the current ranking. This implies that at each phase of our query reformulation simulation, there is a document retrieval step first, followed by the proper generation of simulated queries. There are two phases in our model, as depicted in Figure 7.1: "layer 1" and "layer 2". Layer 1 begins with a real user query and a ranked list of results, which are used to generate simulated possible "next" queries. In layer 2, each of these simulated queries are used to retrieve documents, which in turn are used to generate a second set of simulated possible next queries.

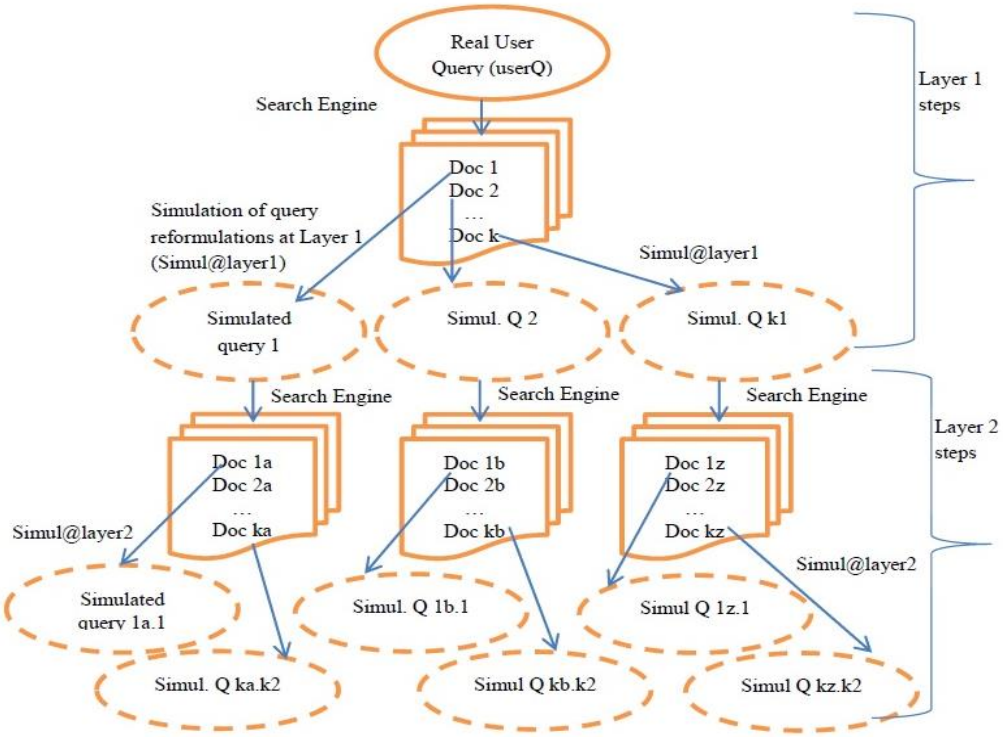


Figure 7.1: A somewhat simple model for generating search history data: The elliptical shapes with large dashes represent the generated simulated queries

7.1.2 A Somewhat More Complex Model

Our second model is a little more complex (see Figure 7.2). It starts the same as the previous model, using a user query and ranked results to generate simulated queries. In addition, the user query is submitted to the general web to obtain a ranking of URLs with snippets. Rather than use the simulated queries to retrieve documents at layer 2, we extract keyphrases from the snippets of the web results to use for document retrieval. Then, as in the original model, a second set of simulated queries are generated from these retrieved documents. Thus the models differ only in the source of queries used to rank documents at layer 2.

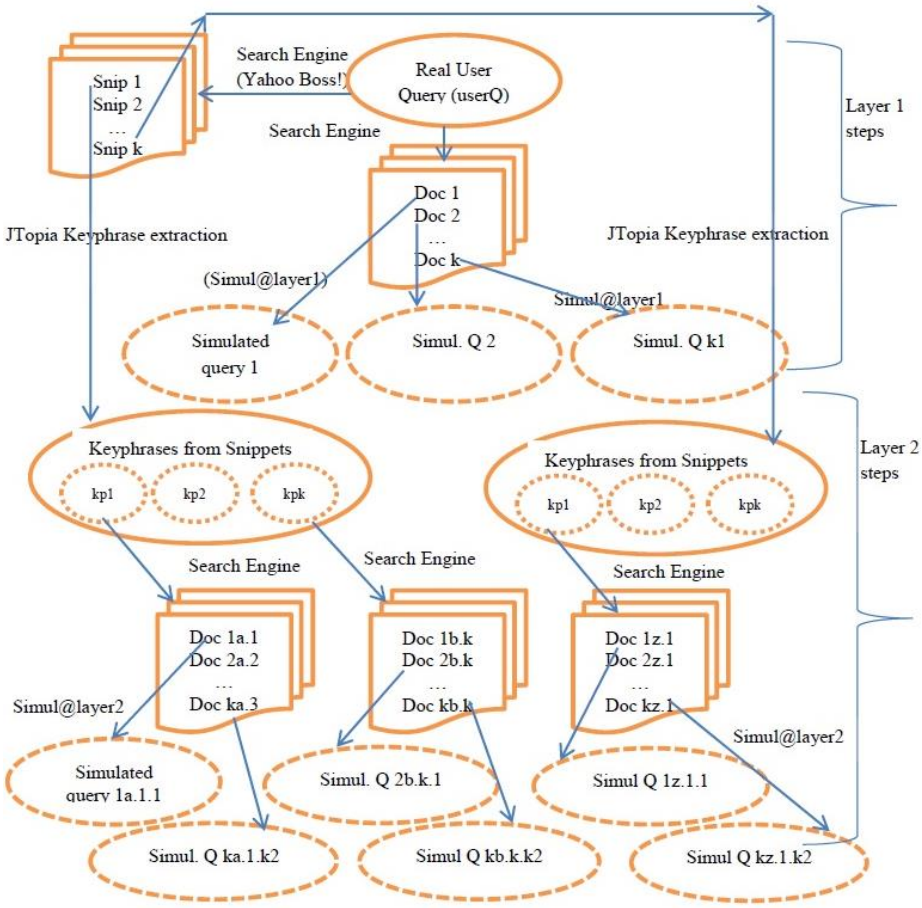


Figure 7.2: A somewhat more complex model for simulating session search data: The elliptical shapes with large dashes represent the generated simulated queries. The elliptical shapes with dashes represent the keyphrases generated in the new steps (they can also be used as simulated queries)

7.2 Implementation of the Methods

To implement our models, we need a retrieval engine and methods for generating simulated queries from ranked results. Below we describe three different implementations of each of the two models.

7.2.1 Layer 1 simulated query reformulations

At layer 1, models 1 and 2 are identical. For the search engine, we use either Indri [94], which uses Dirichlet-smoothed language model scoring to rank full-text documents, or Yahoo! BOSS [105], which returns a SERP with URLs, titles, and snippets. When we use Indri, we extract keyphrases from each of the top-10 full-text documents using JTopia [57], then concatenate the top keyphrases from each document to form a simulated query. When we use BOSS, we select either titles or snippets from the top-10 ranked URLs to be used as simulated queries.

7.2.2 Layer 2 simulated query reformulations

After layer 1, we have 10 simulated queries from one of three possible implementations. At layer 2, the models diverge. For model 1, we essentially repeat layer 1 for each of the top-4 of the 10 simulated queries: the simulated query is submitted to the same search engine, and a new round of simulated queries are generated in the same way. The only difference is that for the BOSS results, we use fewer ranked documents (5 instead of 10).

Model 2 differs by using the original user query a second time, submitting it to Yahoo! BOSS (and only Yahoo! BOSS, not Indri) to obtain snippets of top-ranked documents from the general web. We use JTopia to extract keyphrases from those snippets, and then, unlike model 1 (which uses simulated queries resulting from layer 1), we submit those keyphrases to our engine of choice. Simulated queries are generated from the resulting ranked documents in the same way as in layer 1.

For each of the implementations of layer 1 and layer 2 simulations, we also experiment with a variant in which each simulated query is concatenated with the original query. In this way we guarantee that the simulated query contains the user's

original query, potentially helping to mitigate cases where the simulated query does not contain any of the original query terms.

Additionally, we experiment on the effects of aggregating the ranking resulting from the user's query with the rankings resulting from simulated queries.

7.3 Experiments and Results

7.3.1 Dataset and Evaluation Measures

We use the Session track 2013 dataset [32]. It contains several user sessions which contain one or more interactions. Each interaction consists of a query related to a given information need, a ranked list of results from a search engine, user clicks on the results, and the time spent by the user reading the clicked document. Finally, there is a "current query", the last query in the search session. The 2013 data consists of 87 sessions.

For the effectiveness measure, we adopted the official primary measure used by the TREC Session track organizers, namely nDCG@10. nDCG is a graded relevance measure that rewards documents with high relevance grades and discounts the gains of documents that are ranked at lower positions [53].

7.3.2 Effectiveness of Simulated Queries: Leveraging Simulated Queries

We leverage our generated simulated queries by applying the CombCAT rank fusion method introduced in Chapter 4 [6]. For each query formulation, each top-k ranked document is placed into different bins such that documents that appeared in n different rankings are put in the same bin, labeled "category$_n$". Documents are then reranked in decreasing number of rankings they appeared in. Each simulated query

was submitted to Indri for document retrieval. We compare to the baseline of simply submitting the original user query to the Indri retrieval engine.

7.3.3 Results

Table 7.1: Results for layer 1 of both models on Session track 2013 dataset. Q0x, Q1x, Q2x, Q3x and Q4x respectively denote incorporating the real user query ranking 0, 1, 2, 3 and 4 times in the set of rankings that are being aggregated. QN denotes the concatenation of the real user query to the simulated query.

	Q0X		Q1X		Q2X		Q3X		Q4X	
Resources	ndcg	%Δ	ndcg	%Δ	ndcg	%Δ	ndcg	%Δ	ndcg	%Δ
baseline	0.1147	0.00%								
Jtopia	0.0746	-34.96%	0.0810	-29.38%	0.1371	19.53%	0.1374	19.79%	0.1376	19.97%
QNJTopia	0.1007	-12.21%	0.1017	-11.33%	0.1207	5.23%	0.1216	6.02%	0.1204	4.97%
Titles	0.1465	27.72%	0.1520	32.52%	0.1504	31.12%	0.1474	28.51%	0.1468	27.99%
QNTitles	0.1420	23.80%	0.1331	16.04%	0.1349	17.61%	0.1330	15.95%	0.1338	16.65%
Snip	0.1407	22.67%	0.1596	39.15%	0.1533	33.65%	0.1473	28.42%	0.1459	27.20%
QNSnip	0.1614	40.71%	0.1620	41.24%	0.1573	37.14%	0.1565	36.44%	0.1577	37.49%

7.3.3.1 Can we improve effectiveness at all by simulating queries and leveraging them?

The results in Table 7.1 show that by applying layer 1 simulations alone, we are able to improve the results over the baseline. The highest improvements occur when we leverage Q1X+QNSnip (41.24% improvement over the baseline) and QNSnip (40.71% improvement over the baseline). It is to be noted that although using JTopia alone leads to a significant decrease in effectiveness, using it in addition to user query leads to relatively large improvement (19.53%, 19.79% and 19.97% improvements respectively for Q2x+Jtopia, Q3x+Jtopia and Q4x+Jtopia).

We would also like to compare to a stronger baseline that uses real search history. These results are given in Table 7.2. We can see that using layer 1 alone is

not competitive with using real session history, nor is the first model with both layers. However, using the second model with both layers gives a substantial improvement over using real session data in all cases but the QNSnip method. This suggests that the second model is more than good enough to substitute real session history in the absence of no/little real session history data.

7.3.3.2 When we aggregate the ranking resulting from the user's query with the rankings resulting from simulated queries, does it affect the results?

The results in Table 7.1 suggest that there is generally a positive impact when we aggregate the ranking resulting from the user's query (userQ) with the rankings resulting from leveraging layer 1 simulated queries, as can be seen by comparing results across rows. However the results start degrading when we start over-representing the userQ rankings.

For instance, in the case of Snip, leveraging Snip+Q1x leads to a performance increase of 13.43% over leveraging Snip only (from 0.1407 to 0.1596). This means that including the ranking resulting from the actual user query (only once) helps improve the result by 13.43% over the effectiveness of simply leveraging Snip. But, including those results two times (Snip+Q2x), three times (Snip+Q3x), or four times as much voting rights as the simulated query (Snip) leads to 8.96%, 4.69%, or 3.70% increases over using Snip only.

We conclude that we obtain better results by aggregating the ranking resulting from the user's query with the rankings resulting from leveraging layer 1 simulated queries once, but in most cases only once (except notably for JTopia and QNJTopia).

7.3.3.3 Does concatenating the simulated queries with the original query impact the results?

Comparing the QN variants in Table 7.1 clearly suggests that, in general, concatenating the original query to the simulated query improves the results. For instance, when we go from using Snip to using QNSnip, the results improve from 0.1407 to 0.1614 (14.71% improvement). Results improve by 34.99% from JTopia to QNJTopia. It is worth noting that when leveraging Titles, the nDCG went down from 0.1465 to 0.1420. But that negative change is negligible (3.07% decrease) compared to the 14.71% and 34.99% increase.

7.3.3.4 Is there any added value in going down to layer 2 and deeper?

Table 7.2 shows that, for our first model, layer 2 provides no benefit and in fact hurts effectiveness. This was somewhat foreseeable, in that the queries generated in layer 2 are drifting further away from the original intent.

The second model, however, benefits greatly from the addition of the second layer. Using layer 1 results as strong baselines for the purpose of comparison, the increases in effectiveness from layer 1 to "Complex L1+L2" are in fact 29.22%, 34.30%, 31.49% and 18.59% respectively for Titles, QNTitles, Snip, QNSnip.

Table 7.2: Comparing Layer1 to "Simpler L1+L2" as well as "Complex L1+L2"

Resources	L1 ndcg	Simpler L1+L2 ndcg	%Δ	Complex L1+L2 ndcg	%Δ	Real search history ndcg	%Δ
Titles	0.1465	0.1085	-25.94%	**0.1893**	29.22%	0.1598	9.08%
QNTitles	0.1420	0.1265	-10.92%	**0.1907**	34.30%	0.1722	21.27%
Snip	0.1407	0.1323	-5.97%	**0.1850**	31.49%	0.1715	21.89%
QNSnip	0.1614	0.1326	-17.84%	0.1914	18.59%	**0.1963**	21.62%

7.4 Summary of initial simulation attempt

Here we address the problem of simulating a user who is reformulating queries based on terms and keyphrases s/he encountered during the search process, in order to obtain data similar to search history data that studies leverage for improved effectiveness. In the current study, we assumed a real user provides one single query and nothing else prior to that event, and proposed ways to simulate and generate such data that can be considered to be similar to search history data given that they provide results similar to the ones we obtain when we leverage real users' search history.

7.5 Pilot Study: Towards more human-like simulations of query reformulations

A lot of the simulated queries generated in the initial attempt, described above, are full sentences or very short paragraphs (i.e. titles or snippets). In an effort to generate query reformulations that look even more like the ones users/humans typically enter (i.e. less verbose), we conducted a pilot study in which we proposed another query reformulation model that uses topical language models to generate and filter a space of possible queries, updating language models based on results seen earlier in the session.

7.5.1 Framework and Data

The simulated users (i.e. query simulation modules) simulate queries of the types actual users might provide. The search engine receives queries from a simulated user. Then it returns ranked results to the simulated user, and in return the simulated user simulates interactions, a decision to abandon or not, and a query reformulation.

We have selected a fixed set of 30 topics (a sample of the TREC 2014 Session track topics [31]) for which to generate sessions. We will use the TREC term run, which typically means the ranked results for every topic in a set, for the ranked results

for every query in every session on every topic in the set. The simulation side can run multiple user simulations at the same time to generate M sessions per topic, so a final run will consist of ranked results for each query in each of the M sessions on each of the 30 topics.

The simulated users (i.e. query simulation modules) generate the following information:

1) a first query for each session on each topic – with a fixed set of 30 topics, there will always be 30xM queries for the first round of retrieval.
2) after receiving ranked lists of document IDs for queries:
    (a) titles, URLs, and snippets for the top-10 ranked documents for each
    (b) clicks and dwell times on ranked documents;
    (c) for each session, a decision to stop the session or continue;
    (d) for each continuing session, a query reformulation.

7.5.2 Model

To simulate these query reformulations, we use a two-phase process by which we first generate queries by sampling from a language model, then score them based on their discriminative power among topics. The language "model" we use is actually a series of binomial models with parameters $P(w \in Q|T,I, S_{\{1...i-1\}})$, i.e. the probability of a term w being in a query Q given information about a topic T, the current point in the session i (a discrete number from i - 1 to i_{max}), and the history of the session up to that point $S_{\{1...i-1\}}$.

A key component of our model is that it models query length conditional on topic. Some topics lend themselves more naturally to longer queries. Topic #2, for instance, is looking for information about Dulles International Airport in Washington,

DC. Dulles is an extremely important term for queries on this topic; it appears in almost every real user query. Airport is also an important term appearing in most user queries. Because most queries will contain those two terms by default, queries on this topic should include 3 – 4 terms to find the specific information requested in the topic description (nearby hotels, parking, shuttles to the airport, buses, etc.). For topic #10, on arguments for instituting a tax on "junk food", most queries include the phrase "junk food tax" and therefore have minimum length of 3.

Therefore our model starts by marginalizing over query lengths (we suppress the subscript on S for space reasons):

$$P\left(w \in Q \middle| T, I, S_{\{1\ldots i-1\}}\right) = \sum_{l=1}^{l_{max}} P(w \in Q \middle| T, i, S_{\{1\ldots i-1\}}, l) P(l|T)$$

7.5.2.1 Topical Language Model

The basis of our query generation model will be a binomial model for term presence in queries of length *l* for a topic, that is, $P(w \in Q|T, l)$ independent of the session variables. The maximum-likelihood estimate of this probability is simply the number of times the term appears in topic-related queries in the training data divided by the total number of queries in that data.

$$P_{ML}(w \in Q|T, l) = \frac{qf_{w,l,T}}{qc_{l,T}}$$

Here $qf_{w,l,T}$ is the query frequency, the number of queries of length l term w appears in for topic T, and $qc_{l,T}$ T is the total number of queries of length l recorded for topic T.

These binomial distributions could only be used to generate queries that recombine terms in previously-seen queries. To make it possible to generate queries

using “new” terms (not seen in our source of user queries), we smooth the maximum-likelihood model with additional text data.

For the first query in the session (i = 1), we augment using terms in the topic description shown to users before they begin their search. The rationale for this is that empirically, many of the terms users use in their queries also appear in the topic description (this could be because they use the topic description for guidance, or because the topic descriptions tend to include many of the terms users would naturally use – the direction of causality is not clear).

$$P(w \in Q|T, l, i = 1) = \frac{qf_{w,l,T} + \mu \frac{to_{w,T}}{|T_{desc}|}}{qc_{l,T} + \mu}$$

where $to_{w,T} = 1$ if term w appears in the topic description of T and $to_{w,T} = 0$ otherwise, and $|T_{desc}|$ is the number of unique terms in the topic description.

For each subsequent query (i > 1), we include more information about the session history in the model. In particular, terms from titles and snippets of documents seen by the user in previous results:

$$P(w \in Q|T, l, i) = \frac{qf_{w,l,T} + \mu \frac{sf_{w,T}}{|S|}}{qc_{l,T} + \mu}$$

Here $sf_{w,T} = 1$ if term w appears in the topic description of T, or in the title of any document retrieved by the previous query (the one at i – 1), or in the snippet of any document retrieved by the previous query, and |S| is the number of unique terms in titles and snippets retrieved.

Note that this still limits query generation to terms seen in real user queries (which will have high probability) and terms seen in topic descriptions and titles/snippets of retrieved documents (which will have much lower probability distributed uniformly across unique terms).

7.5.2.2 Sampling Queries

First we sample a query length ` from a distribution P(L|T) derived from training data. Then we iterate over terms in the language model in order of decreasing probability, flipping a coin to determine whether to add the term or not, until we have a query of length l.

Table 7.3: Top 4 most-frequent terms appearing in queries for three topics with their binomial occurrence model probability P (w ∈ Q), their multinomial language model probability P (w|Q), and their frequency in queries sampled using the procedure in Section 7.5.2.2.

		in training data		sampled
topic	term	P (w ∈ Q)	P (w|Q)	P (w ∈ Q')
2	dulles	0.95	0.29	1.00
	airport	0.95	0.29	0.99
	hotel	0.13	0.04	0.17
	metro	0.12	0.04	0.19
11	milestone	0.71	0.14	0.87
	culture	0.66	0.12	0.85
	development	0.61	0.11	0.87
	infant	0.44	0.07	0.55
48	evaluate	0.98	0.28	1.00
	employee	0.93	0.27	0.95
	to	0.30	0.09	0.10
	how	0.19	0.06	0.00

Note that this procedure gives greater probability for the highest-frequency terms in real queries to appear in sampled queries. Essentially we boost the probabilities of the most common terms above what they would be otherwise. This is meant to mitigate a problem with language models, that their probabilities tend to

underestimate the importance of common terms while overestimating the importance of rare terms. Table 7.3 shows examples of terms in our topics, their binomial model probabilities, their multinomial language model probabilities, and their frequency of occurrence when sampling using our approach.

At each step i of the session, we generate N candidate queries. One query will be sampled from the set to be returned as the simulated reformulation.

7.5.2.3 Scoring Sample Queries

After generating N candidates, each one is scored according to the probability that each word in the query could generate the topic that the query is meant for. This is a way of scoring candidate queries by their ability to discriminate among the topics.

$$P(T|w) = \frac{qtf_{w,T} + stf_{w,T} + \mu \frac{1}{|T|}}{qf_w + stf_w + \mu}$$

where $qtf_{w,T}$ is the total number of times w appears in queries on topic T, $qt_{f,w}$ is the total number of times w appears in all queries on any topic, and |T| is the total number of unique topics. $st_{f,w,T}$ and $st_{f,w}$ are the "session frequencies" of the term (in topic and across topics, respectively), and include the counts of the term in the topic description and in titles and snippets of documents retrieved for previous queries.

Each candidate query Q_j is scored as:

$$P(Q_j) = \prod P(w|T) \propto \prod P(T|w)P(w)$$

where P(w) is a prior probability of term w, which for now we treat as uniform.

The scores are then renormalized into a proper probability distribution, i.e.

$$P_{norm}(Q_j|T) = \frac{P(Q_j|T)}{\sum_{k=1}^{N} P(Q_k|T)}$$

and one candidate is sampled from this distribution to be the simulated reformulation.

7.5.2.4 Session abandonment

At this stage of the simulation development, we do not model "abandonment" in the sense of a user stopping a session due to success or frustration or any other reason real users might decide to stop. We simply sample a session length from a distribution t to existing user data.

Let i_{max} be the length of a session (that is, the number of rounds of querying in the session). We will define:

$$P(i_{max}) = \frac{\#\ of\ session\ length\ i_{max}}{\#\ of\ sessions}$$

To model "abandonment", we simply sample a session length from this distribution, and the session will stop after that many rounds of querying.

7.5.2.5 Summary of the model

We have described a two-stage approach by which queries are first generated from a series of binomial distributions, then scored using a topical discriminator. The first part generates candidates that are practically guaranteed to contain the most important terms and phrases (as observed in real user queries), while the second part ensures that there will still be variety in other terms included in queries.

We use TREC 2013 and 2014 Session track data to fit distributions $P(i_{max})$, $P(l|T)$ for each topic T in our set, $P(w \in Q|l, T)$ for each query length for each topic T. The full steps are as follows:

1. For each topic T:
    a. Sample i_{max} and loop from i = 1 to i_{max}:
        i. Update $P(w \in Q|l, T, i)$ using text in the topic description and results for query i – 1
        ii. For j = 1 to N:

A. Sample a query length

B. Sample a query Q_j by sampling 1/0 from $P(w \in Q|l,T,i)$ until l terms sampled

C. Score query by $P(T|Q_j)$

iii. Sample one query Q_{sim} from $P(Q|T)$ to send to search engine

iv. Receive retrieval results from the search engine

b. Loop back to step (a) to simulate another session for the same topic

We note that instead of using our two-stage process, we could sample queries directly from a traditional multinomial language model. However, the queries such a model generates tend to not look much like real queries: rare terms appear too frequently, terms that frequently appear together in real queries rarely occur together in sampled queries, etc. Some of these problems could be resolved using n-gram models and better smoothing, but our more heuristic approach seems to work well.

7.5.3 Experiments and Results

Finally we turn to evaluating simulated queries. There is an aspect of human judgment to it (i.e. do the queries look like queries a person would enter?) but it may not be the most important factor. We care most about whether the queries we generate are "good" for evaluating retrieval systems, and in particular, whether they are good for evaluating systems that use features derived from session history.

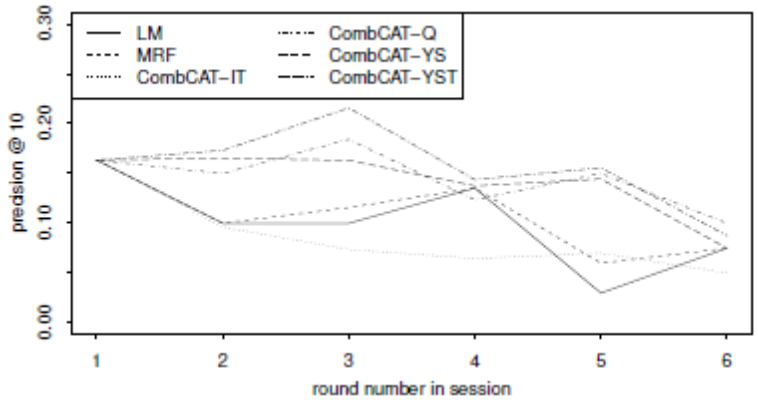


Figure 7.3: Comparison of six retrieval systems across six rounds of a session using non-simulated queries

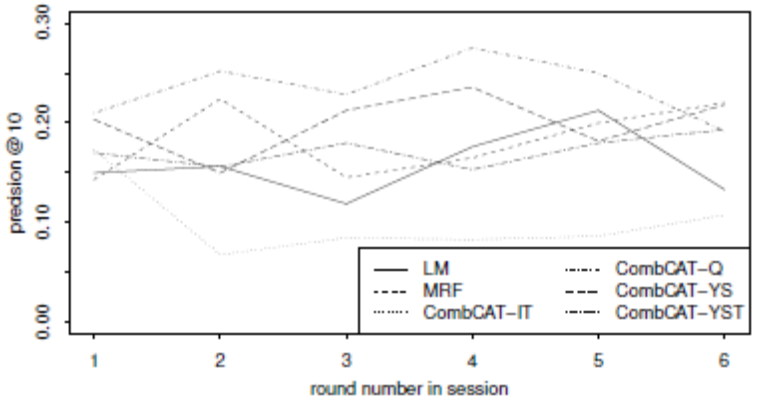


Figure 7.4: Comparison of six retrieval systems across six rounds of a session using simulated queries

To that end, in this section we will evaluate simulated queries indirectly by evaluating retrieval systems that take them as input. All of our systems are based on Indri and an index of ClueWeb12 that has been filtered using spam scores from the Waterloo spam classifier [43]. Two of them (LM and MRF) are ad hoc systems that treat each query as an independent event and do not make any use of session history.

Four of them (CombCAT-*) fuse different sources of data derived from the session history that emanated from the query simulations' results; these use the CombCAT fusion method that has been successful in the TREC Session track and which we introduced in Chapter 4 [6], [6b]. This method takes strings of text from various sources in the session history, uses those strings as a query, then fuses the results from all strings based on the frequency with which documents occur.

One of the query simulation modules running was providing user queries sampled uniformly from those submitted for the Session track; since they were being sampled randomly they are not based on session history. We refer to this as *non-sim*.

Another model was the one described in Section 7.4.2, which uses terms from previously-ranked documents to generate queries. We refer to this as *sim*.

We evaluate all runs by precision@10 using TREC Session track relevance judgments.

We look at the ability of simulated queries to rank systems by relative effectiveness. Our goal is to determine whether simulated queries can distinguish between systems of different effectiveness over the session more efficiently than non-simulated queries.

Table 7.4: Overall precision@10 averaged across all sessions and all rounds in each session for each of our six systems.

Rank	Run	non-sim P10	Run	sim P10
1	CombCAT-YST	0.1566	CombCAT-Q	0.2345
2	CombCAT-Q	0.1452	CombCAT-YS	0.2003
3	CombCAT-YS	0.1415	MRF	0.1830
4	MRF	0.1082	CombCAT-YST	0.1721
5	LM	0.1001	LM	0.1579
6	CombCAT-IT	0.0863	CombCAT-IT	0.1001

Table 7.4 presents averages of the precision@10 numbers shown in Figure 7.3 across all six rounds in the session. The rank correlation between the two is 0.6, mainly because the CombCAT-YST drops from rank 1 with non-simulated queries to rank 4 with simulated queries.

We note the following:

- the fact that simulated queries produce higher precision@10 is evident in these two plots;
- the differences between systems are more pronounced with simulated queries than with non-simulated queries;
- the same system is more-or-less consistently at the bottom in both cases: CombCAT-IT, which uses titles of previously-ranked documents as queries, then fuses results for the current round.
- the baseline LM ad hoc system is consistently among the lower-performing systems in both cases;
- the top system is not the same: for non-simulated queries, CombCAT-YST appears better, while for simulated queries CombCAT-Q performs better.

Overall, our conclusion is that non-simulated queries and simulated queries provide similar overall session evaluation results.

It is difficult to say that one ranking is more "correct" than the other, but we note that the differences between systems are larger with simulated queries than with non-simulated queries. This suggests that even if non-simulated and simulated queries agree on the relative ordering of systems, using simulated queries magnifies the differences.

Table 7.5: Examples of sequences of queries for six topics in our set. For each topic we show an actual user session of queries and a simulated session

topic	query type	queries
2	non-sim	dulles airport → dulles airport location → dulles hotels
2	sim	airport dulles hotels stop → airport dulles park → airport cheap dulles stop → airport dulles hotels metro→ airport dulles metro near → airport dulles hotels metro
3	non-sim	jobs from business phds → business phd
3	sim	benefits business cost master → benefits business phd → business mba phd→ business cost master phd→ benefits business phd worth → business doctoral phd
11	non-sim	infants development culture→ infant development "cultural effects" → infant OR child development intitle:culture → infant OR child development milestones→ infant OR child development milestones research
11	sim	culture developmental infant milestones → culture infant milestones → infant milestones → culture developmental infant milestones → culture developmental milestones
15	non-sim	internet phone services → internet phone services review → guide internet phone services → voip providers
15	sim	providers reviews voip → services voip → cheapest internet phone services → providers reviews voip→ features voip → providers reviews voip
48	non-sim	employee evaluation → evaluate employees
48	sim	employee evaluation → employee evaluation performance → employee evaluation guide→ employee evaluation
60	non-sim	malaria impact on economy africa → malaria economy africa → malaria economy → malaria donations
60	sim	africa aids charity hiv malaria → charity hiv → aids charity → africa aids charity → charity hiv → africa aids charity ght hiv

Finally, we return to the question of whether our simulated sessions "look like" real user sessions. Table 7.5 shows some examples of real user sessions and simulated sessions for a random selection of our 30 topics.

Note that simulated queries often use very similar terms and phrases as actual user queries (though sometimes the ordering of terms in a phrase is lost in the simulated queries). Topic 11 shows an example of a user trying different possible features of a query language such as phrasing with quotes, boolean OR, and looking for a term in the title; none of this can be captured by the simulation as it currently exists.

We do not draw any broad conclusions from this table; we only show it to give a sense of what our simulated queries and sessions look like compared to actual sessions.

7.6 Summary

In this chapter, we strived to generate data similar to search history data that retrieval models for search over session, such as the one we presented in Chapter 4, leverage for improved effectiveness. We addressed the problem by simulating a user who reformulates queries based on terms and keyphrases s/he encountered during the search process. We assumed that a real user provides one single query and nothing else prior to that event, and proposed ways to simulate and generate data that can be considered to be "similar" to search history data. We concluded that our generated query reformulations are similar to real search history data because they provide results as similarly effective as the ones we obtain when we leverage real users' search history. We furthered our study by proposing to generate shorter and less verbose

simulated queries that look more like the kind of query reformulations real users are likely to provide to a search engine. This query reformulation model used topical language models to generate and filter a space of possible queries, updating language models based on results seen earlier in the session.

Chapter 8

CONCLUSION AND FUTURE WORK

This work is an effort with the ultimate goal of building more effective and robust retrieval systems for complex search tasks in situations where there are only small amounts of search history data. We started this thesis by conducting an experiment aimed at understanding users' preferences when examining results returned for their broad queries which are typical in the context of complex search tasks. Our focus was to investigate their preferences with respect to the comprehensiveness of a document and the relevance grade of the document. We achieved this by using the so-called triplet framework via Amazon Mechanical Turk to empirically show that users tend to prefer in large proportions documents with high aspect coverage, regardless of the topical relevance grade. When asked to choose between two documents D_1 and D_2, the one that is most useful for learning more about the topic, given a prior document D_T, users overwhelmingly preferred comprehensive documents (i.e. documents with highest subtopic coverage). In fact, they preferred comprehensive documents even when the prior document D_T already covers more subtopics than each of D_1 and D_2. Likewise, even in cases where D_1 and D_2 are relevant to the same number of novel subtopics, the one that is relevant to the largest overall subtopics was most often preferred.

Using those results as a guiding principle, we introduced a heuristic data fusion retrieval model that tends to prioritize documents that are likely to be relevant to the largest number of subtopics. We simply gauged this by assuming that a document is

potentially relevant to a larger number of subtopics than another document if it appears in more rankings than the other. In addition to that simple heuristic data fusion retrieval model – that we called CombCAT – we used two existing data fusion methods, CombSUM and CombMNZ. We empirically showed that using Bing's related queries is a good choice, but using session-dependent data is even better. Furthermore we found that we achieved even better results by combining Bing related queries to session-dependent queries.

In an effort to build retrieval models that are not only effective, but also robust, we conducted an experiment wherein our retrieval systems strive to mitigate situations where the improved systems fare worse than the baseline on certain queries. We proposed two re-ranking approaches based on exploiting document "popularity" across a topic, and show that these methods can help improve average overall effectiveness as well as robustness. Using the runs submitted to TREC Web track 2013 and 2014 as baselines, we show that, after our re-ranking, overall effectiveness gets improved in an overwhelming number of cases, and robustness gets improved in a large number of cases but fewer than for overall effectiveness. Our future efforts on robustness-aware systems may focus on establishing a principled framework for better exploiting the "popularity" of documents as well as other features to improve robustness of systems.

Given that our retrieval models thus far are heuristic, and considering that retrieval models explicitly derived on principles such as probability theory or linear algebra have the advantage of building on well-understood definitions and theorems, as well as providing a more general framework to build on; we proposed a Probabilistic Data Fusion framework (PDF) which derives on the well-known

Probability Ranking Principle (PRP). PDF, makes a small amendment to the PRP function by suggesting to rank documents in decreasing order of their probability of relevance and retrieval by systems, as opposed to relevance only. We proposed a way to estimate the probability that documents are retrieved by exploiting various sources of sample possible queries. We proceeded to show the impacts of different choices for several model components by implementing different instances of our model and empirically show that, when used with very rich sources of sample possible queries, they are at least on-par with the best reported systems for different search scenarios including ad-hoc search, diversity search and search over sessions. Specifically, we show them to be at least competitive with the best reported systems for TREC Session track 2013 and 2014, TREC Web track 2013 and 2014 as well as NTCIR IMine 2014 dataset. Future work for this investigation will involve experimenting with different various values for each component. For instance, we will experiment with a query sampling probability that uses language model-style smoothed generation to allow the estimation for very short queries. We may try probabilities that account for training based on clicked documents, or that simply increase the voting rights of a possible query q' by the number of documents that were clicked when the ranking for q' was displayed to the user. We will also investigate the impact of using other discount functions for P(retr|q', D), for instance a logarithmic discount (like nDCG) or a geometric discount (RBP).

Finally, our last effort consisted in generating data similar to search history data by simulating query reformulations, in the absence of query reformulations. This is important because in some situations, there is little to no search history data available to be leveraged by the system. Our approach was to simulate a user who is

reformulating queries based on terms and keyphrases s/he encountered during the search process, in order to obtain data similar to search history data that studies leverage for improved effectiveness. We assumed that a real user provides one single query and nothing else prior to that event, and proposed ways to simulate and generate such data that can be considered to be similar to search history data given that they provide results similar to the ones we obtain when we leverage real users' search history. Furthermore, in an effort to generate query reformulations that look even more like the ones users typically enter (i.e. less verbose), we conducted a pilot study in which we proposed another query reformulation model that uses topical language models to generate and filter a space of possible queries, updating language models based on results seen earlier in the session.

From the experiments conducted in Chapters 4 and 6, we can conclude the following about the effectiveness of our retrieval framework:

- Overall, our data fusion retrieval framework works well when we leverage session-dependent data: As can be seen in Table 4.2, for 2011 TREC Sessions dataset, we get 23% ndcg@10 increase over the baseline when we leverage most significant key-phrases from previously retrieved top documents; for 2012 Sessions dataset, we get 28% improvement when we leverage snippets; for 2013 and 2014 respectively, we get peaks of 12% and 15% improvement when we leverage. However, as we explained in section 6.3.1.1, we note that there is underperformance in several settings when we exploit titles for the 2014 dataset, and that the difference between 2014 dataset and the other datasets can be explained by differences in the engine that

generated search results, differences in the user population, and differences in the amount of data.

- One source of related queries that consistently works well across all datasets is external query logs: As can be seen in Table 6.1, for 2014 and 2013 TREC Sessions datasets respectively, we get 36% and 40% ndcg@10 increase over the baseline when we leverage external snippets; on the same datasets, we obtain 27% and 8% improvements when we leverage Bing suggestions. As can be seen in Table 4.4, for 2012 and 2011 TREC Sessions datasets respectively, we get 19% and 4% ndcg@10 increase over the baseline when we leverage Bing suggestions.
- Additionally, we found that combining different sources and exploiting them as related queries tends to bring upon better improvements than exploiting individual sources. For the 2011 and 2012 Sessions data respectively, we reach peaks of 36% and 42.5% ndcg@10 increase when we combine Bing and most significant keyphrases.
- Finally, another conclusion about the effectiveness of our retrieval framework is that it is very helpful to leverage information from other users who searched the same topic. For example, on the 2014 and 2013 datasets respectively, we get 78% and 66% improvements over the baseline systems which are much larger than the improvements of 36% and 40% respectively that we obtain without leveraging other users' search sessions.

REFERENCES

1. AlchemyAPI. http://www.alchemyapi.com. 2014.

2. Alzghool, M., & Inkpen, D. (2010). A novel class-based data fusion technique for information retrieval. *Journal of Emerging Technologies in Web Intelligence*, *2*(3), 160-166.

3. Amazon mechanical turk. http://www.mturk.com. 2012.

4. Aslam, J. A., & Montague, M. (2001, September). Models for metasearch. In Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval (pp. 276-284). ACM.

5. Aslam, J., Callan, J., Manmatha, R., Sanderson, M., & Voorhees, E. (2002). Metasearch: Data fusion and distributed retrieval. In *Workshop on Challenges in Information Retrieval and Language Modeling*.

6. Bah, A., & Carterette, B. (2014, December). Aggregating results from multiple related queries to improve web search over sessions. In *Asia Information Retrieval Symposium* (pp. 172-183). Springer International Publishing.

7. Bah, A., & Carterette, B. (2015, December). Improving Ranking and Robustness of Search Systems by Exploiting the Popularity of Documents. In *Asia Information Retrieval Symposium* (pp. 174-187). Springer International Publishing.

8. Bah, A., & Carterette, B. (2016, October). Fusing Search Results from Possible Alternative Queries. To Appear In *2015 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*.

9. Bah, A., & Carterette, B. (2016, September). Generating Pseudo Search History Data in the Absence of Real Search History. In *International Conference on Database and Expert Systems Applications* (pp. 410-417). Springer International Publishing.

10. Bah, A., & Carterette, B. (2016, September). PDF: A Probabilistic Data Fusion Framework for Retrieval and Ranking. In *Proceedings of the 2016 ACM on International Conference on the Theory of Information Retrieval* (pp. 31-39). ACM.

11. Bah, A., Carterette, B., & Chandar, P. (2014). Udel@ NTCIR-11 IMine Track. In *NTCIR*.

12. Bah, A., Chandar, P., & Carterette, B. (2015, August). Document Comprehensiveness and User Preferences in Novelty Search Tasks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 735-738). ACM.

13. Bah, A., Sabhnani, K., Zengin, M., & Carterette, B. (2014). University of delaware at TREC 2014. In TREC. 2014.

14. Bartell, B. T., Cottrell, G. W., & Belew, R. K. (1994, August). Automatic combination of multiple ranked retrieval systems. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 173-181). Springer-Verlag New York, Inc..

15. Baskaya, F. (2014). Simulating Search Sessions in Interactive Information Retrieval Evaluation. Tampere University Press.

16. Baskaya, F., Keskustalo, H., & Järvelin, K. (2011, April). Simulating simple and fallible relevance feedback. In *European Conference on Information Retrieval* (pp. 593-604). Springer Berlin Heidelberg.

17. Baskaya, F., Keskustalo, H., & Järvelin, K. (2012, August). Time drives interaction: simulating sessions in diverse searching environments. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval* (pp. 105-114). ACM.

18. Belkin, N. J., Kantor, P., Fox, E. A., & Shaw, J. A. (1995). Combining the evidence of multiple query representations for information retrieval. *Information Processing & Management*, 31(3), 431-448.

19. Bendersky, M., Metzler, D., & Croft, W. B. (2010, February). Learning concept importance using a weighted dependence model. In *Proceedings of the third ACM international conference on Web search and data mining* (pp. 31-40). ACM.

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

21. Bhattacharjee, R., & Goel, A. (2007, January). Algorithms and incentives for robust ranking. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms* (pp. 425-433). Society for Industrial and Applied Mathematics.

22. Bing. http://www.bing.com. 2014.

23. Blair, D. C., & Maron, M. E. (1985). An evaluation of retrieval effectiveness for a full-text document-retrieval system. *Communications of the ACM*, 28(3), 289-299.

24. Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., & Hullender, G. (2005, August). Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning* (pp. 89-96). ACM.

25. Büttcher, S., Clarke, C. L., & Lushman, B. (2006, August). Term proximity scoring for ad-hoc retrieval on very large text collections. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 621-622). ACM.

26. Carbonell, J., & Goldstein, J. (1998, August). The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 335-336). ACM.

27. Carterette, B., & Chandar, P. (2009, November). Probabilistic models of ranking novel documents for faceted topic retrieval. In *Proceedings of the 18th ACM conference on Information and knowledge management* (pp. 1287-1296). ACM.

28. Carterette, B., Bah, A., & Zengin, M. (2015, September). Dynamic Test Collections for Retrieval Evaluation. In *Proceedings of the 2015 International Conference on The Theory of Information Retrieval* (pp. 91-100). ACM.

29. Carterette, B., Bennett, P. N., Chickering, D. M., & Dumais, S. T. (2008, March). Here or there. In *European Conference on Information Retrieval* (pp. 16-27). Springer Berlin Heidelberg.

30. Carterette, B., Kanoulas, E., & Yilmaz, E. (2011, October). Simulating simple user behavior for system effectiveness evaluation. In *Proceedings of the 20th ACM international conference on Information and knowledge management* (pp. 611-620). ACM.

31. Carterette, B., Kanoulas, E., Hall, M., & Clough, P. (2014). *Overview of the TREC 2014 session track.* In TREC.

32. Carterette, B., Kanoulas, E., Hall, M., Bah, A., & Clough, P. (2013). Overview of the TREC 2013 session track. In TREC.

33. Chandar, P., & Carterette, B. (2012). What qualities do users prefer in diversity rankings. In *Proceedings of the 2nd Workshop on Diversity in Document Retrieval*.

34. Chandar, P., & Carterette, B. (2012, August). Using preference judgments for novel document retrieval. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval* (pp. 861-870). ACM.

35. Chapelle, O., Ji, S., Liao, C., Velipasaoglu, E., Lai, L., & Wu, S. L. (2011). Intent-based diversification of web search results: metrics and algorithms. *Information Retrieval*, 14(6), 572-592.

36. Chapelle, O., Metlzer, D., Zhang, Y., & Grinspan, P. (2009, November). Expected reciprocal rank for graded relevance. In *Proceedings of the 18th ACM conference on Information and knowledge management* (pp. 621-630). ACM.

37. Chen, S. F., & Goodman, J. (1996, June). An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics* (pp. 310-318). Association for Computational Linguistics.

38. Clarke, C. L., Craswell, N., & Soboroff, I.: Overview of the trec 2012 web track. In TREC (2012)

39. Clarke, C. L., Kolla, M., Cormack, G. V., Vechtomova, O., Ashkan, A., Büttcher, S., & MacKinnon, I. (2008, July). Novelty and diversity in information retrieval evaluation. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 659-666). ACM.

40. Collins-Thompson, K. (2009, November). Reducing the risk of query expansion via robust constrained optimization. In *Proceedings of the 18th ACM conference on Information and knowledge management* (pp. 837-846). ACM.

41. Collins-Thompson, K., Bennett, P., Diaz, F., Clarke, C. L., & Voorhees, E. M.: Trec 2013 web track overview. In TREC 2013.

42. Collins-Thompson, K., Bennett, P., Diaz, F., Clarke, C. L., & Voorhees, E. M.: Trec 2014 web track overview. In TREC 2014.

43. Cormack, G. V., Smucker, M. D., & Clarke, C. L. (2011). Efficient and effective spam filtering and re-ranking for large web datasets. *Information retrieval*, 14(5), 441-465.

44. Dang, V., & Croft, B. W. (2010, February). Query reformulation using anchor text. In *Proceedings of the third ACM international conference on Web search and data mining* (pp. 41-50). ACM.

45. Diaz, F., & Metzler, D. (2006, August). Improving the estimation of relevance models using large external corpora. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 154-161). ACM.

46. Dinçer, B. T., Ounis, I., & Macdonald, C. (2014, April). Tackling biased baselines in the risk-sensitive evaluation of retrieval systems. In *European Conference on Information Retrieval* (pp. 26-38). Springer International Publishing.

47. Dou, Z., Hu, S., Luo, Y., Song, R., & Wen, J. R. (2011, October). Finding dimensions for queries. In *Proceedings of the 20th ACM international conference on Information and knowledge management* (pp. 1311-1320). ACM.

48. Fox, E. A., & Shaw, J. A. (1994). Combination of multiple searches. *NIST SPECIAL PUBLICATION SP*, 243-243.

49. Garofolo, J. S., Auzanne, C. G., & Voorhees, E. M. (2000). The TREC Spoken Document Retrieval Track: A Success Story. *NIST SPECIAL PUBLICATION SP*, 500(246), 107-130.

50. Guan, D. (2013). Structured Query Formulation and Result Organization for Session Search (Doctoral dissertation, Georgetown University).

51. Guan, D., Zhang, S., & Yang, H. (2013, July). Utilizing query change for session search. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval* (pp. 453-462). ACM.

52. Hiemstra, D. (1998). A linguistically motivated probabilistic model of information retrieval. In *Research and advanced technology for digital libraries* (pp. 569-584). Springer Berlin Heidelberg.

53. Järvelin, K., & Kekäläinen, J. (2002). Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems* (TOIS), 20(4), 422-446.

54. Järvelin, K., Price, S. L., Delcambre, L. M., & Nielsen, M. L. (2008, March). Discounted cumulated gain based evaluation of multiple-query IR sessions. In *European Conference on Information Retrieval* (pp. 4-15). Springer Berlin Heidelberg.

55. Jiang, J., He, D., & Han, S. (2012). On Duplicate Results in a Search Session. In TREC.

56. Jiang, J., He, D., Han, S., Yue, Z., & Ni, C. (2012, October). Contextual evaluation of query reformulations in a search session by user simulation. In *Proceedings of the 21st ACM international conference on Information and knowledge management* (pp. 2635-2638). ACM.

57. JTopia. https://github.com/srijiths/jtopia. 2015

58. Kang, C., Wang, X., Chen, J., Liao, C., Chang, Y., Tseng, B., & Zheng, Z. (2011, February). Learning to re-rank web search results with multiple pairwise features. In *Proceedings of the fourth ACM international conference on Web search and data mining* (pp. 735-744). ACM.

59. Kanoulas, E., Carterette, B., Clough, P. D., & Sanderson, M. (2011, July). Evaluating multi-query sessions. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval* (pp. 1053-1062). ACM.

60. Kanoulas, E., Carterette, B., Clough, P. D., Sanderson, M.: Overview of the trec 2011 session track. In TREC'11 (2011)

61. Kanoulas, E., Carterette, B., Clough, P. D., Sanderson, M.: Overview of the trec 2012 session track. In TREC (2012)

62. Keskustalo, H., Järvelin, K., Pirkola, A., Sharma, T., & Lykke, M. (2009, October). Test collection-based IR evaluation needs extension toward sessions–a case of extremely short queries. In *Asia Information Retrieval Symposium* (pp. 63-74). Springer Berlin Heidelberg.

63. Kruschwitz, U. (2012). University of Essex at the TREC 2012 Session Track. In TREC.

64. Lavrenko, V., & Croft, W. B. (2001, September). Relevance based language models. In Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval (pp. 120-127). ACM.

65. Lebanon, G., & Lafferty, J. (2002, July). Cranking: Combining rankings using conditional probability models on permutations. In *ICML* (Vol. 2, pp. 363-370).

66. Lee, J. H. (1995, July). Combining multiple evidence from different properties of weighting schemes. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 180-188). ACM.

67. Lee, J. H. (1997, July). Analyses of multiple evidence combination. In *ACM SIGIR Forum* (Vol. 31, No. SI, pp. 267-276). ACM.

68. Lillis, D., Toolan, F., Collier, R., & Dunnion, J. (2006, August). Probfuse: a probabilistic approach to data fusion. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 139-146). ACM.

69. Liu, T. Y. (2009). Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3), 225-331.

70. Liu, Y., Song, R., Zhang, M., Dou, Z., Yamamoto, T., Kato, M. P., ... & Zhou, K. (2014). Overview of the NTCIR-11 IMine Task. In *NTCIR*.

71. Luhn, H. P. (1957). A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of research and development*, 1(4), 309-317.

72. Lv, Y., & Zhai, C. (2009, July). Positional language models for information retrieval. In Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval (pp. 299-306). ACM.

73. Lv, Y., & Zhai, C. (2010, July). Positional relevance model for pseudo-relevance feedback. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval* (pp. 579-586). ACM.

74. Lv, Y., Zhai, C., & Chen, W. (2011, July). A boosting approach to improving pseudo-relevance feedback. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval* (pp. 165-174). ACM.

75. Metzler, D., & Croft, W. B. (2005, August). A Markov random field model for term dependencies. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 472-479). ACM.

76. Miller, D. R., Leek, T., & Schwartz, R. M. (1999, August). A hidden Markov model information retrieval system. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 214-221). ACM.

77. Monge, A., and Elkan, C. (1996, August). The field-matching problem: algorithm and applications. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*. (pp. 267-270). ACM.

78. Montague, M., & Aslam, J. A. (2002, November). Condorcet fusion for improved retrieval. In *Proceedings of the eleventh international conference on Information and knowledge management* (pp. 538-548). ACM.

79. Ng, K. B., & Kantor, P. B. (2000). Predicting the effectiveness of naive data fusion on the basis of system characteristics. *Journal of the American Society for Information Science*, 51(13), 1177-1189.

80. Ponte, J. (2001). Is information retrieval anything more than smoothing. In *Proceedings of the Workshop on Language Modeling and Information Retrieval* (pp. 37-41).

81. Ponte, J. M., & Croft, W. B. (1998, August). A language modeling approach to information retrieval. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 275-281). ACM.

82. Radlinski, F., & Dumais, S. (2006, August). Improving personalized web search using result diversification. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 691-692). ACM.

83. Raman, K., Bennett, P. N., & Collins-Thompson, K. (2013, July). Toward whole-session relevance: exploring intrinsic diversity in web search. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval* (pp. 463-472). ACM.

84. Robertson, S. E. (1977). The probability ranking principle in IR. *Journal of documentation*, 33(4), 294-304.

85. Robertson, S. E., Walker, S., Beaulieu, M., & Willett, P. (1999). Okapi at TREC-7: automatic ad hoc, filtering, VLC and interactive track. *Nist Special Publication SP*, 253-264.

86. Robertson, S. E., Walker, S., Jones, S., Hancock-Beaulieu, M. M., & Gatford, M.: Okapi at TREC-3. In TREC. 1994.

87. Salton, G., Wong, A., & Yang, C. S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, 18(11), 613-620.

88. Santos, R. L., Macdonald, C., & Ounis, I. (2010, April). Exploiting query reformulations for web search result diversification. In *Proceedings of the 19th international conference on World wide web* (pp. 881-890). ACM.

89. Schutz, A. T. (2008). Keyphrase extraction from single documents in the open domain exploiting linguistic and statistical methods. *Dissertation, National University of Ireland*.

90. Shaw, J. A., & Fox, E. A. (1995). Combination of multiple searches. *NIST SPECIAL PUBLICATION SP*, 105-105.

91. Shokouhi, M., White, R. W., Bennett, P., & Radlinski, F. (2013, July). Fighting search engine amnesia: Reranking repeated results. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval* (pp. 273-282). ACM.

92. Sparck Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1), 11-21.

93. Sparck Jones, K., Walker, S., & Robertson, S. E. (2000). A probabilistic model of information retrieval: development and comparative experiments: Part 2. *Information Processing & Management*, 36(6), 809-840.

94. Strohman, T., Metzler, D., Turtle, H., & Croft, W. B. (2005, May). Indri: A language model-based search engine for complex queries. In *Proceedings of the International Conference on Intelligent Analysis* (Vol. 2, No. 6, pp. 2-6).

95. Tao, T., & Zhai, C. (2007, July). An exploration of proximity measures in information retrieval. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 295-302). ACM.

96. Vogt, C. C., & Cottrell, G. W. (1998, August). Predicting the performance of linearly combined IR systems. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 190-196). ACM.

97. Vogt, C. C., Cottrell, G. W., Belew, R. K., & Bartell, B. T. (1996). Using Relevance to Train a Linear Mixture of Experts. In *TREC* (Vol. 5, No. 2, pp. 503-515).

98. Walker, S., Robertson, S. E., Boughanem, M., Jones, G. J., & Jones, K. S. (1997, November). Okapi at TREC-6 Automatic ad hoc, VLC, routing, filtering and QSDR. In *TREC* (pp. 125-136).

99. Wang, J., & Zhu, J. (2009, July). Portfolio theory of information retrieval. In Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval (pp. 115-122). ACM.

100. Wang, L., Bennett, P. N., & Collins-Thompson, K. (2012, August). Robust ranking models via risk-sensitive optimization. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval* (pp. 761-770). ACM.

101. Wu, S. (2009). Applying statistical principles to data fusion in information retrieval. *Expert Systems with Applications*, *36*(2), 2997-3006.

102. Wu, S., & McClean, S. (2006). Improving high accuracy retrieval by eliminating the uneven correlation effect in data fusion. *Journal of the American Society for Information Science and Technology*, 57(14), 1962-1973.

103. Wu, S., & McClean, S. (2006). Performance prediction of data fusion for information retrieval. *Information processing & management*, 42(4), 899-915.

104. Wu, S., Bi, Y., & Zeng, X. (2011, January). The linear combination data fusion method in information retrieval. In *Database and Expert Systems Applications* (pp. 219-233). Springer Berlin Heidelberg.

105. Yahoo ! Boss. https://developer.yahoo.com/search/boss/. 2015

106. Yang, Y., & Lad, A. (2009, September). Modeling expected utility of multi-session information distillation. In *Conference on the Theory of Information Retrieval* (pp. 164-175). Springer Berlin Heidelberg.

107. Yue, Z., Jiang, J., Han, S., & He, D. (2012, October). Where do the query terms come from?: an analysis of query reformulation in collaborative web search. In *Proceedings of the 21st ACM international conference on Information and knowledge management* (pp. 2595-2598). ACM.

108. Zhai, C. X., Cohen, W. W., & Lafferty, J. (2003, July). Beyond independent relevance: methods and evaluation metrics for subtopic retrieval. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval* (pp. 10-17). ACM.

109. Zhai, C., & Lafferty, J. (2001, September). 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* (pp. 334-342). ACM.

110. Zhai, C., & Lafferty, J. (2002, August). Two-stage language models for information retrieval. In Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval (pp. 49-56). ACM.

111. Zhang, S., Guan, D., & Yang, H. (2013, July). Query change as relevance feedback in session search. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval* (pp. 821-824). ACM.

112. Zhu, J., Wang, J., Cox, I. J., & Taylor, M. J. (2009, July). Risky business: modeling and exploiting uncertainty in information retrieval. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval* (pp. 99-106). ACM.