The Role of Document Structure and Citation Analysis in Literature Information Retrieval

A Thesis

Submitted to the Faculty

of

Drexel University

by

Haozhen Zhao

in partial fulfillment of the

requirements for the degree

of

Doctor of Philosophy

September 2015

Dedications

In memory of my grandfather

Jinbang Zhao

(1937—1997)

who ignited my intellectual curiosity when I was a child.

To my parents

Zhongyong Zhao and Cailuan Wang

for their unconditional love and support in every stage of my life.

To my grandmother, my maternal grandparents and my sisters

for their love.

Acknowledgments

This dissertation would be impossible without the constant support and effective guidance of Dr. Xiaohua (Tony) Hu. I am most grateful for Dr. Hu's help. I appreciate Dr. Yuan An, Dr. Weimao Ke, Dr. Erjia Yan and Dr. Li Sheng of being my committee.

I appreciate my dear friend Dr. Forest Woody Horton Jr. for treating me as a family member since I came to the US.

Over my seven years at the iSchool PhD program, I am helped by many professors and friends. I am much obliged to them for their kindness and friendliness, and will never forget their help.

Lastly, I would like to acknowledge my parents, my grandparents, my sisters, my uncles and aunts for their love and support over the past thirty years of my life.

Table of Contents

List of Tables

List of Figures

Abstract

The Role of Document Structure and Citation Analysis in Literature Information Retrieval

Haozhen Zhao

Advisor: Xiaohua (Tony) Hu, Ph.D

Literature Information Retrieval (IR) is the task of searching relevant publications given a particular information need expressed as a set of queries. With the staggering growth of scientific literature, it is critical to design effective retrieval solutions to facilitate efficient access to them. We hypothesize that particular genre specific characteristics of scientific literature such as metadata and citations are potentially helpful for enhancing scientific literature search. We conducted systematic and extensive IR experiments on open information retrieval test collections to investigate their roles in enhancing literature information retrieval effectiveness.

This thesis consists of three major parts of studies. First, we examined the role of document structure in literature search through comprehensive studies on the retrieval effectiveness of a set of structure-aware retrieval models on ad hoc scientific literature search tasks. Second, under the language modeling retrieval framework, we studied exploiting citation and co-citation analysis results as sources of evidence for enhancing literature search. Specifically, we examined relevant document distribution patterns over partitioned clusters of document citation and co-citation graphs; we examined seven ways of modeling document prior probabilities of being relevant based on document citation and co-citation analysis; we studied the effectiveness of boosting retrieved documents with scores of their neighborhood documents in terms co-citation counts, co-citation similarities and Howard White's pennant scores. Third, we combined both structured retrieval features and citation related

features in developing machine learned retrieval models for literatures search and assessed the effectiveness of learning to rank algorithms and various literature-specific features.

Our major findings are as follows. State-of-the-art structure-ware retrieval models though reportedly perform well in known item finding tasks do not significantly outperform non-fielded baseline retrieval models in ad hoc literature information retrieval. Though relevant document distributions over citation and co-citation network graph partitions reveal favorable pattern, citation and co-citation analysis results on the current iSearch test collection only modestly improve retrieval effectiveness. However, priors derived from co-citation analysis outperform that derived from citation analysis, and pennant score for document expansion outperforms raw co-citation count or cosine similarity of co-citation counts. Our learning to rank experiments show that in a heterogeneous collection setting, citation related features can significantly outperform baselines.

Chapter 1: Introduction

1.1 Introduction

The body of scientific literature is growing at a staggering rate. Take the biomedical domain as an example, the literature has been growing at a "double-exponential pace"; both the total size and the number of new papers published each year have a compounded annual growth rate of about 3% to 4%[2]. A quick search on PubMed shows that on average more than 3,000 papers were published per day in 2013. Effective literature search solutions are thus crucial for researchers and professionals to stay on top of the torrent of publications.

Although web search has enjoyed great technological and commercial successes over the past two decades, building effective information retrieval (IR) systems for specific domains is still a challenging task. In specific domains, e.g. scientific literature or patents search, both the corpus to be searched and the queries submitted by the end users often possess particular characteristics that have the potential to be leveraged for effective retrieval. For example, queries in domain specific search may differ from queries in web search, which are typically short and ambiguous.

Scientific Literature Search (SLS) is the task of searching related publications for scholars. Scientific literature here includes online library public access catalog, journal and conference research papers' bibliographic records and their full text, etc. An effective SLS system could facilitate a quick and accurate knowledge access, which is critical for both academia and industry. But the dramatic growing publications have posted serious challenges for efficient literature search.

The importance of effective literature search engine is evident for any researcher or

professional in practice, because researches are built upon previous endeavors in science and in conducting any research a researcher must acquire a good knowledge of the subject at hand through finding and reading important relevant literature.

There are several characteristics of scientific literature.

- Semi-structured: Scientific literature contains many metadata. For example, bibliographic records contain fields such as title, authors, venue information, subject headings, keywords, abstract, description and so on.
- Interconnected: Scientific literature, especially, journal and conference papers contain many references and citations that interconnect them. These connections convey important information about the relation among them.
- Heterogeneous: There are different types/genres of scientific literature and each with their own particular metadata schema, vocabularies, term and corpus statistics.

These characteristics pose both opportunities and challenges to designing effective solutions to facilitate the access to scientific literature.

First, intuitively field information of the scientific literature should be used in enhancing retrieval. In fact, the purpose of some of the metadata fields, e.g. subject heading and keywords, are designed for making access of the resources easy and most of them can be used in designing better browsing access to scientific literature. However, not many existing retrieval models make use of the structure information. Most retrieval models in existing researches take a non-structured view of the document and merge all fields into one. Of the few that are structure-aware, still is unknown of their performance in literature search tasks.

Second, though citation analysis on scientific literature is well developed in the domain

of bibliometrics, how to ingest the insights from bibliometric analysis into building effective retrieval models is an open question. There are various approaches in leveraging citations in facilitating search, but the results are not conclusive. Moreover, very few information retrieval studies paid attention to co-citation analysis, which potentially can be a good source of evidence for retrieving literature.

Third, given the heterogeneous nature of the scientific literature, we need new retrieval framework that can embrace its heterogeneity. Traditional IR models generally use federated search and data fusion techniques to deal with heterogeneous collections, while the recent arising learning to rank framework is powerful enough to include multitude sources of evidences as features and to deliver retrieval functions based on established machine learning techniques. It would be interesting to compare these two paradigms in our scientific literature search scenario.

These observations motivate us to investigate enhancing scientific literature search that leverages structure, citation and learning to rank techniques.

The premise of this thesis is that particular characteristics of scientific literature should be leveraged in building effective IR systems for literature search. Structure and citation information in scientific literature that are not well treated in established IR modeling approaches should be re-examined in contributing evidence for determining the relevance of a document against a query. The goal of this thesis is to enhance literature search with models that capture important aspects of the scientific literature corpus.

Before proceeding with our discussion, we need to clarify the focus of this dissertation. First, we distinguish the two most general ad hoc retrieval tasks related to literature search: known item search and subject search. The known item search task is to search for the documents that the users know their existence in the system, therefore they are also called

look-up search[3]. Examples of known item search include looking up books or papers written by a concerned author in the library catalog system, or searching for homepage of a person or an organization on the Web. The information need is generally met with one or few best candidates. Subject search involves searching for documents related to certain subject/topic, for example, "string theory" in the physics domain. The goal for the IR system is to retrieve as many and as accurate relevant items as possible. In this thesis, we are going to focus on subject search, specifically, keyword-based search over semi-structured interconnected documents, and leaving known item literature search for future work. Second, there is also related work on providing effective literature search though interface-designing innovations. Faceted search is one approach to leverage fields in search and browsing. But there are several drawbacks of faceted search: it is expensive to maintain metadata of high quality; it costs screen real estate, especially when there are many fields; not all fields are suitable for faceted display. This thesis will focus on retrieval algorithm and model perspectives, instead of interface and user studies.

1.2 Research Questions

The overall research question of this dissertation is how to leverage structure and citation information in developing effective retrieval models for searching in heterogeneous scientific literature collection. We approach this question in a "divide and conquer" manner. We first separately deal with developing structure-aware retrieval models for literature search and leveraging citation and co-citation analysis in designing retrieval models. Then we target retrieval solutions in the heterogeneous information space and include findings from the first and the second research questions into an integral learning to rank framework for literature search. Therefore, the overall question is divided into three major research questions:

RQ 1 Can we leverage document structure with structure-aware retrieval models?

Many researches on retrieving structured information have been conducted by the database community and XML retrieval community. The database community generally deals with exact data match, while in IR partial match is the case. Our focus differs from theirs in that (1) we deal with semi-structured information; (2) fields in our semi-structured documents are non-repeatable and non-hierarchical[4]. In the IR community, there are mainly two kinds of approaches in modeling semi-structured document retrieval: (1) small document combination approach, which treats each field as individual small document, and linearly combine their scores as the document score; (2) in-model combination approach, which preserves properties of underlying retrieval model while combining evidences based on fields[5]. Structure-aware retrieval models, such as PL2F, Mixture of Language Model (MLM) and Probabilistic Retrieval Model for Semi-structured Documents (PRMS) have been shown to be effective for in known item finding tasks according to previous reported studies. But there performance in ad hoc literature search tasks is still unknown. Can we leverage the rich structural data of scientific literature with these models? How will different stcuture-aware retrieval models perform in literature search?

RQ 2 How to leverage citation and co-citation analysis information to enhance literature search?

Citation is an integral part of scientific literature and plays an important role in communicating researches. It is assumed that citation has great potential for enhancing literature

search. But how to model them in retrieval frameworks is an open question. How to leverage bibliometric analysis results to enhance search results ranking? How to use models based on the structure and properties of the information space to enhance IR model? Whether there is some favorable pattern in about citation network such that we can use for enhancing literature search? Under the language modeling framework, there are several possible ways to ingest citation analysis results into the retrieval model. Will using language model document prior and selective search strategies help?

RQ 3 How to effectively rank documents in a heterogeneous literature collection?

With features derived from fields and citation analysis, there will be many features for scientific literature. Moreover, of different genres, e.g. catalog, bibliographic records and full-text, documents generally have different sets of features. How to rank document in such a heterogeneous environment is challenging. One approach to deal with different genres or document types is to use the fusion techniques. In a fusion framework, retrieval runs returned from different retrieval strategies, retrieval systems, indices are merged with some algorithm into the final results to present to end users. It is expected that this way the effectiveness of different retrieval strategies can be captured, and each index is optimized according to its own properties, thus the overall retrieval effectiveness will be improved. However, results from a previous study which tried a collection fusion method over the iSearch test collection, indicates that this method does not beat the single index baseline method[6].

With the advent of learning to ranking techniques, it is possible to include more features

in retrieval models. We thus plan to try the learning to rank methods to solve the problem of searching in heterogeneous collections. We will derive a set of features for literature search, and test them with the state of the art learning to rank algorithms. We pay particular attentions to features that are specific to literature search domain, e.g. citation related features, and fields related features.

We want to investigate whether learning to rank approach works for literature search; whether it can capture promising evidences based on structure-aware and bibliometric-enhanced retrieval models.

1.3 Thesis Outline

In this thesis, we study how to use structure-aware retrieval models and citation-aware approaches to enhance literature search. Specifically, we investigate the following aspects of literature search: (1) leverage structure/field information to enhance literature search; (2) leverage citation and co-citation analysis information to enhance retrieval. (3) use learning to rank methods to find out effective features for literature search.

This dissertation is organized as following: Chapter 1 covers the background of this thesis and proposes the research questions, and introduces the datasets that will be used in this dissertation and evaluation methodologies. Chapter 2 discussed basics of information retrieval and review related subject areas. Chapter 3 evaluates structure aware retrieval model on the iSearch test collection. Chapter 4 studies enhancing literature search with citation and co-citation analysis. Chapter 5 investigates learning to rank for literature search. Chapter 6 concludes this dissertation and discusses future directions.

1.4 Contributions

The following contributions are made in this dissertation:

1. We compared the performance of major existing structure-aware retrieval models on literature search tasks exhaustively. We find out that structured-aware retrieval models though reportedly perform well in known item finding tasks do not perform well in ad hoc literature search tasks.

2. We discovered that partitioning scientific literature corpus based on analysis of the citation and co-citation network will be potentially beneficial for deploying selective search strategy which will be more efficient while not necessarily less effective than that of exhaustive search strategy.

3. We empirically studied seven ways of deriving language model document priors based on citation and co-citation analysis, and evaluate their performance on an open IR test collection.

4. We studied several ways of document expansion approaches based on co-citation analysis, and showed that the newly proposed pennant score based similarities outperforms more established similarity measures.

5. We extensively studies the performance of three learning to rank algorithms and a set of structure and citation related features in developing machine learned retrieval models.

1.5 Dataset and Evaluation Methodology

We evaluate our approaches using the iSearch Collection.

1.5.1 The iSearch Collection

The iSearch collection was prepared by the iSearch team. It approximately consists of 18K book MAchine-Readable Cataloging (MARC) records (BK), 291K articles metadata

(PN) and 160K PDF full text articles (PF), plus 3.7 million extracted internal citation entries among PN and PF. 65 topics drawn from physics researchers' real information needs with corresponding relevance judgment data also come with the collection[7]. Of all the PN and PF documents, 259,093 are cited at least once, which is chosen as the subset for our experiment for reducing citation sparseness consideration, we call this subset **PNPFCited** collection. Table 1.1 shows the basic statistics of the iSearch test collection.

Table 1.1: iSearch Test Collection Statistics

Section	Description	Number
BK	Library Records	18,441
PN	Abstracts, arXiv.org	291,244
PF	PDF items, arXiv.org	143,569

There are two separate processing of the iSearch dataset for our experiment. One is the full dataset; the other is a subset, which focus on the investigation of the citation feature. The same subset is also used in our previous work[8], where we keep items that are cited at least once for studying the citation prior. For the subset, accordingly, we removed documents not in our index from the relevance judgment files, then filtered out topics without any relevant documents in the relevance judgment data, resulting 57 valid topics out of the original 65 topics (topic 5, 6, 15, 17, 20, 25, 42, 54, 56 are excluded). The full dataset is used in Chapter 3, 5, and the subset is used in Chapter 4.

1.5.2 Topics and Relevance Judgments

The iSearch dataset come with 65 topics with relevance judgment results, based on 65 natural search tasks (topics) from 23 researchers and students from university physics departments. The topic owners were given a search task description form with five fields:

(a) What are you looking for? (current information need)

(b) Why are you looking for this? (work task)

(c) What is your background knowledge of this topic? (background knowledge)

(d) What should an ideal answer contain to solve your problem or task? (ideal answer)

(e) Which central search terms would you use to express your situation and information need? (search terms)

An example of the iSearch topic is shown in Figure 1.1.

```
<topic_id>002</topic_id>
<author_id>085</author_id>
<current_information_need>
  I am looking for information about manipulation and sorting of
  magnetic particles, beads or spheres on nanoscale. This might be in
  a micro fluidic system.
</current_information_need>
<work_task>
  As a part of my master thesis it is interesting to fabricate a
  sorting device which can sort magnetic nano spheres from a sample.
  This will often be in a micro fluidic device because the nano
  sphere/particles often will be diluted in some sort of solution.
</work_task>
<background_knowledge>
  I have been making sorting devices for micro particles based on flow
  profiles in a microfluidic system.
</background_knowledge>
<ideal_answer>
  Published material on how to sort magnetic beads, particles or
  spheres on nanoscale.
</ideal_answer>
<search_terms>
  Nano spheres, beads, magnetic, sorting
</search_terms>
```

Figure 1.1: Example topic of the iSearch collection, number 002

Table 1.2 gives the distribution of the iSearch relevance judgment (qrels) dataset.

1.5.3 iSearch Citation and Co-citation Network

The iSearch citation network contains 82% of all the PNs and 97% of all the PFs. On average each item in PN has 32.2 citations and each item in PF has 36.1 citations.

We also processed the iSearch collection to generate the paper co-citation network of

Table 1.2: Distribution of relevant documents over tasks[1]

Range of relevant docs	No. of tasks (N=65)
>100	9
75-100	3
50-74	8
25-49	13
15-24	12
10-14	8
<10	12

Table 1.3: Descriptive Statistics of Relevant Documents across the iSearch Document Types

	BK	PN	PF	All
Total	424	1078	1376	2878
Mean	6.4	16.3	20.8	43.6
Var	6.3	20.9	30.8	47.7
Median	5	8	7	24.5

the iSearch collection. We calculated the document co-citation counts and compiled all the co-citation among the indexed papers, resulting a weighted undirected graph with 259,093 vertices and 33,888,861 edges, with edge weights being the number of times two papers are cited together.

1.5.4 Evaluation Metrics

IR systems generally can be evaluated in terms of effectiveness and efficiency. Since the Cranfield studies set up the paradigm of IR evaluation, majority of IR studies are concerned with effectiveness evaluation. In the Cranfield paradigm, IR evaluation setup consists of a set of topics, a document collection and a set relevance judgments. Most widely used retrieval performance evaluation measures include Precision@k, MAP, and NDCG.

Given a function R defined as $R(i) = 1$ if the document at rank i is relevant and $R(i) = 0$

otherwise, then Precision@k is

$$Precision@k = \frac{\sum_{i \leq k} R(i)}{k} \tag{1.1}$$

Average Precision (AP) measures the average precision after each relevant document is retrieved

$$AP = \frac{\sum_{k:R(k)=1} Precision@k}{|R|} \tag{1.2}$$

where $|R|$ denotes the total number of relevant documents in the result set. Mean Average Precision (MAP) is AP averaged over all topics.

$$MAP = \frac{\sum_{q \in Q} AP(q)}{|Q|} \tag{1.3}$$

AP and MAP are set-based retrieval evaluation metrics as they take a binary view of relevance, without distinguishing a highly relevant document from a marginally relevant document. Discounted cumulative gain (DCG) is a metric proposed in , which allows graded relevance. In DCG, it is assumed that relevant documents at a low rank should be discounted by their rank. DCG is calculated as with the following formula:

$$DCG(k) = \sum_{j \leq k} G(j) \cdot N(j) \tag{1.4}$$

where $G(\cdot)$ is the gain function and $N(\cdot)$ is the normalization function. Let rel_i denotes the graded relevance value at position i of the result list, we define the gain function and discount function respectively as:

$$G(i) = 2^{rel_i} - 1 \tag{1.5}$$

$$N(i) = \frac{1}{\log_2(i+1)} \tag{1.6}$$

Then $DCG@k$ is:

$$DCG@k = \sum_{i=1}^{k} \frac{2^{rel_i} - 1}{\log_2(i+1)} \tag{1.7}$$

There are two ways of defining DCG, with minor difference. Definition in Equation 1.7 is often used in reported learning to rank studies.

In IR research, $DCG@k$ is further normalized by an ideal $DCG@k$, $IDCG@k$, which is the maximum $DCG@k$ in all possible permutation of the k results. The normalized DCG, $NDCG@k$ is defined as:

$$NDCG@k = \frac{DCG@k}{IDCG@k} \tag{1.8}$$

Previous work on the iSearch Collection was generally evaluated with the NDCG@1000[1]. The best performed retrieval model is language model with Jelinek-Mercer smoothing.

BPREF, binary preference, is a retrieval effectiveness measure when the relevance judgments are incomplete[9]. Given a topic has R relevant documents and r is a relevant document and n is a document from the set of judged non relevant documents that are ranked higher than r, then $bpref$ is given as Equation 1.9. BPREF correlates with AP when the relevant judgments are complete and is more robust than AP when the relevant judgments are incomplete as it penalizes judged non relevant document ranking higher than judged relevant

[1]This is the default implementation in `trec-eval`, `http://trec.nist.gov/trec_eval/`.

document regardless of retrieving unjudged documents.

$$bpref = \frac{1}{R}\sum_{r}(1 - \frac{|n\ ranked\ heigher\ than\ r|}{R}) \tag{1.9}$$

Expected Reciprocal Rank is a new measure proposed in . It is argued that NDCG does not take use's effort into consideration. In DCG, each document has a constant gain which is independent to other documents in the search result list. ERR is used in learning to rank for training rankers. Previous researches shows that trained on one measure, e.g. ERR, then test on another e.g. NDCG, can result good results. Given P_i is the probability of a user satisfied with the i-th search result,

$$P_i = \frac{2^{rel_i} - 1}{2^{rel_{\max}}} \tag{1.10}$$

where $rel_{\max}$ is the maximum relevance grade, $ERR@k$ is computed as

$$ERR@k = \sum_{i=1}^{k} \frac{P_i}{i} \prod_{j=1}^{k-1}(1 - P_j) \tag{1.11}$$

All retrieval results in this thesis, except those in Chapter 5, are evaluated using the standard IR evaluation toolkit `trec_eval` from NIST. Performance scores in Chapter 5 are calculated with the internal evaluation functions implemented in RankLib.

In IR, comparison two retrieval setup is also tested for significance. In this dissertation, we use pair Student t test, from SciPy implementation, to ascertain if the difference between two retrieval settings is significant or not.

Chapter 2: Literature Review

In this chapter, we review the basics of information retrieval to establish the background for this thesis work. We first review the basics of information retrieval. Then we review existing structured information retrieval work, citation based IR work, and learning to rank researches.

2.1 Information Retrieval

The goal of an information retrieval system is to find information that meets the end user's information need. Broadly information retrieval is defined as "a field concerned with the structure, analysis, organization, storage, searching, and retrieval of information" [10]. As a domain of which the major goal is to help users find information their want [11]. From a schematic perspective, every information retrieval system consists of three components: collection to be indexed, the user's request, generally keywords, and the matching algorithm. The effectiveness of the IR system depends on better understanding of the user's information need. At its core, IR is about modeling relevance. Information retrieval systems are essentially based on the underlying retrieval models.

In a typical information retrieval system evaluation environment, there are user information need and their queries, search corpus or collections and the relevance judgment results.

$$P(R, Q, D) = P(R|Q, D)P(Q, D) \tag{2.1}$$

2.2 Statistical Language Model

Language modeling approach started when the probabilistic retrieval approach was proposed. In this model, the ranking problem is turned into an estimation problem. In this model, both the user's query and the documents are treated as language models. In this model, the purpose is to estimate an accurate query language model and document language model.

Given a query from the user, the goal of an IR system is to rank returned documents as accurately as possible such that the user's information needs will be satisfied. To achieve this goal, we need to design a retrieval model that can capture the query and document relationship effectively, such that relevant documents are delivered while non-relevant documents are avoid at best. Over the years, many different IR models have been developed. The key concept in a retrieval model is relevance. IR models differ in how they formalize the concept of relevance. For example, in the vector space retrieval model, query and documents are represented as term vectors over the vocabulary space, and the relevance between a query and a document is modeled as the distance between their term vectors. In other words, the more similar a document to a query, the more relevant it is to the query.

The probabilistic retrieval model takes a difference approach; it directly models on relevance by representing relevance as a binary-valued event [12;13]. The Probabilistic Ranking Principle (PRP), justified in [14], underpins the probabilistic retrieval model. The PRP prescribes that optimal retrieval effectiveness is achieved when documents are ranked in decreasing order of probability of relevance or usefulness to the request and "probabilities are estimated as accurately as possible on the basis of whatever data has been made available on to the system" [14]. This implies that it is possible to go beyond document text and incorporate any evidence that might be helpful to improve the retrieval effectiveness in a

principled way. Strong probabilistic and statistical foundations make probabilistic retrieval models powerful to address complex retrieval problems.

Statistical language model is the recent generation of probabilistic retrieval models [15;16]. A language model is a probability distribution over sequences of words. Each document can have its own language model. If we regard a query as a sample from a document language model, then we will desire to rank documents based on the probabilities of generating the query using their language models. The retrieval task then becomes to "infer a document model for each document, then [to] estimate the probability of generating the query from each of these models" [17].

Thus the score of document against a query in the language model is based on the conditional probability $P(D|Q)$, given Q, D represent the query and the document respectively. Using the Bayes' rule, we can derive $P(D|Q)$ in the following way:

$$P(D|Q) = \frac{P(Q|D)P(D)}{P(Q)} \tag{2.2}$$

$$\propto P(Q|D)P(D) \tag{2.3}$$

$$\propto \prod_{w} P(w|D)P(D) \tag{2.4}$$

One reason for this derivation is that in practice $P(Q|D)$ is usually easier to estimate and implement than $P(D|Q)$. Note that in the above equations, we dropped the P(Q) because it does not depend on the document thus not affect the ranking. With this derivation, we get two important components of the language model $P(Q|D)$, the query likelihood, and $P(D)$, the document prior. This type of language model therefore is also known as the query likelihood language model.

If we take independence assumption of query term occurrence, the query likelihood $P(Q|D)$ can be further rewritten as $\prod_{w \in Q} P(w|D)$. Now $P(w|D)$ is determined by the distribution of the terms in the document. In Ponte's paper, multiple Bernoulli distribution is used[17]. Nowadays multinomial word distribution is more often used. With this distribution, the query likelihood language model is also called unigram language model. In this model, $P(w|D)$ is equal to proportion of their occurrence count to the document length, with the maximum likelihood estimation:

$$P(w|D) = \frac{c(w, D)}{|D|} \tag{2.5}$$

where $c(w, D)$ denotes the count of word w in document D and $|D|$ is the length of D.

The document prior, $P(D)$, is often assumed to be uniform for all documents and thus not affect the ranking, and can be dropped when scoring the document. However, in fact, $P(D)$ provides an elegant manner to incorporate non-textual evidences into the language model. Some successful applications of $P(D)$ in particular retrieval tasks include: using URL type as a source of $P(D)$ for named page finding[18], using document quality as document prior[19], using temporal evidence as document priors for retrieval in newswire collection[20], etc.

2.2.1 Language Model Smoothing

One problem lies in the query likelihood model is that for a multiple-word query when the document does not contain one word from the query, $P(Q|D)$ will become zero, even when it contains all the rest of the query words. This behavior is undesirable and should be avoided. Many smoothing techniques are developed to avoid the problem of zero probabilities for unseen terms, though the role of smoothing include both for assigning proper weights for

unseen terms and improve the discriminant power of some elite terms[21]. We review here some representative smoothing methods.

The Jelinek-Mercer smoothing interpolates the document language model with a collection language model:

$$P_\lambda(w|D) = (1-\lambda)P_{ml}(w|D) + \lambda P(w|C) \tag{2.6}$$

where $P_{ml}(w|D)$ and $P(w|C)$ are the maximum likelihood estimation of the document model and collection model, respectively, and λ controls the relative weights of the two models.

The Bayesian smoothing using Dirichlet priors, or simply, Dirichlet smoothing, is a smoothing method that set the conjugate prior the multinomial distributed document model using a Dirichlet distribution with the following parameters related to the collection model

$$(\mu P(w_1|C), \mu P(w_2|C), \ldots, \mu P(w_n|C)) \tag{2.7}$$

and the document model is then

$$P_\mu(w|D) = \frac{c(w;D) + \mu P(w|C)}{|D| + \mu} \tag{2.8}$$

where $c(w;D)$ is the term frequency of w in document D, and $|D|$ is the document length. The intuition behind the Dirichlet smoothing is that long documents should be smoothed less than short documents.

The statistical translation model proposed by[22] can be also regarded as a smoothing method. In this model, the query is treated as a translation of the document by all the

words in the document:

$$P_t(w|D) = \frac{|D|}{|D|+\mu} \sum_{\mu \in D} P(w|u)P(u|D) + \frac{\mu}{|D|+\mu} P(w|C) \tag{2.9}$$

where $P(w|\mu)$ is the probability of translating word u to v. With this, it is possible to score a document with non-zero probability even when w does not occur in D, as long as there are words in D, which are semantically related to w.

The semantic smoothing method proposed by [23;24] is a smoothing method inspired by the translation language model. It leverages topic signatures, which are meaningful multi-word phrases or ontology concepts, in smoothing $P(w|D)$:

$$\begin{aligned} P(w|D) &= \lambda P_t(w|D) + (1-\lambda)P(w|C) && (2.10) \\ &= \lambda \sum_k P(w|t_k)P(t_k|D) + (1-\lambda)P(w|C) && (2.11) \end{aligned}$$

where t_k is the topic signature that co-occurs with term w, $P(t_k|D)$ is maximum likelihood estimation of the probability of generating t_k from a document

$$P(t_k|d) = \frac{c(t_k, D)}{\sum_i c(t_i, D)} \tag{2.12}$$

and $P(w|t_k)$, the translation probability from topic signature t_k to term w, is estimated using an Expectation Maximization (EM) algorithm [25]. Specifically, they account the occurrence of term w with a mixture of topic signature t_k's language model and the background

collection language model,

$$P(w|\theta_{t_k}, C) = \lambda P(w|\theta_{t_k}) + (1-\lambda)P(w|C) \tag{2.13}$$

and maximize the log likelihood of generating document sets associated with t_k, D_k,

$$\log P(D_k|\theta_{t_k}, C) = \sum_w c(w, D_k) \log P(w|\theta_{t_k}, C) \tag{2.14}$$

2.2.2 Kullback-Leibler Divergence Language Model

One limitation of the query likelihood language model is that it cannot handle relevance feedback and query expansion in a principled way. The Kullback-Leibler (KL) divergence language model is proposed to address this problem[26]. KL divergence is a measure originated from information theory, which measures the distance between two probability distributions. If we use θ_Q and θ_D to denote the query and document word distributions, i.e., query and document language models, respectively. The KL-divergence of θ_Q from θ_D, denoted $D_{KL}(\theta_Q \| \theta_D)$, measures the information lost when θ_Q is used to approximate θ_D[27]. In IR, it is desirable that the less information is lost the better. Therefore we should rank documents using negative KL divergence:

$$\begin{aligned}
Score(Q, D) &= -D_{KL}(\theta_Q \| \theta_D) && (2.15)\\
&= -\sum_{w \in Q} P(w|\theta_Q) \log \frac{P(w|\theta_Q)}{P(w|\theta_D)} && (2.16)\\
&= \sum_{w \in Q} P(w|\theta_Q) \log p(w|\theta_D) - \sum_{w \in Q} P(w|\theta_Q) log P(w|\theta_Q) && (2.17)\\
&\propto \sum_{w \in Q} P(w|\theta_Q) \log P(w|\theta_D) && (2.18)
\end{aligned}$$

$\sum_{w \in Q} P(w|\theta_Q) \log P(w|\theta_Q)$ in the above equations does not affect document ranking, therefore can be dropped in speeding up the scoring process. With this formalization, now we can do both relevance feedback and document expansion in language model. In the case of relevance feedback, we need to come up with a better query language model, $\hat{\theta_Q}$, which can be, as an example, interpolated with an expanded query model θ_F:

$$\hat{\theta_Q} = (1 - \lambda)\theta_Q + \lambda\theta_F \tag{2.19}$$

where λ controls the relative weights between the original query language model and the relevance feedback model. In the case of document expansion, the goal is to develop a better document language model $\hat{\theta_D}$.

The Relevance Model use a different way to deal with relevance feedback[28].

2.2.3 Cluster-based Retrieval Model

Cluster-based retrieval models are founded on the famous Cluster Hypothesis which states that "closely associated documents tend both to belong to the same clusters and to be relevant to the same requests"[29?] . Justifications of using clustering in language models as shown in previous studies include: (1) similar information need under the query can be met with similar documents[30]; (2) corpus structure should be a good source of evidence for smoothing documents[31]; (3) using good neighborhood documents can solve the insufficient document sampling issue[32].

Liu and Croft proposed to use cluster to smooth language model[33]. They interpolate the document language model with a cluster language model:

$$P(w|D) = \lambda P_{ML}(w|D) + (1-\lambda)P(w|Cluster) \quad (2.20)$$

$$= \lambda P_{ML}(D) + (1-\lambda)[\beta P_{ML}(w|Cluster) + (1-\beta)P_{ML}(w|Coll)] \quad (2.21)$$

where $P(w|Cluster)$ is the cluster language model and $P_{ML}(w|Cluster)$ is its maximum likelihood estimation. They further studied three ways of representing a cluster: concatenating all member documents of a cluster into a large document; term frequency (TF) mixture representation consisting of a weighted mixture of term frequencies of member documents; document model (DM) mixture consisting of a weighted mixture of document language models of member documents[34]. Of the three methods, DM mixture method performs the best.

Other than being used for smoothing language model, cluster can also be used as a retrieval strategy. It is based on the observation that often a large percentage of relevant documents belong to some query-specific clusters. If we can retrieve these optimal clusters and then either interactively improve it based on user's feedback or automatically do a local re-ranking, the retrieval effectiveness will be increased[35].

Citations indicate strong relationships among documents, and have been used in clustering documents[36]. They are potentially helpful for both the cluster-based smoothing and cluster-based retrieval tasks. However, based on our knowledge, no previous research has experimented with citation-based clustering for either of the tasks. This may due to that datasets used in previous related studies usually do not carry citation information. In literature search task domain, citations is an important part and abundant. Therefore, it will be very interesting to examining citation's role in these tasks for literature search.

2.3 Semi-structured Retrieval Models

In most retrieval models, documents are represented as bags of words, omitting the structure of the document. One way to deal with semi-structured documents is to simply merge words from fields as a single document. But that fails to exploit the structure information of the document, which is potentially useful. In the early Text REtrieval Conference (TREC) tasks, researchers found that retrieval effectiveness can be improved when multiple representations of the document or collection are combined in a post retrieval fusion approach[37]. This brought about a set of data fusion methods, such as CombSum[37], CombMNZ[37], Condercet[38], etc., which fuse either the score result or the rank result of multiple runs over different document representations or retrieval algorithms. The first retrieval approach that deal with semi-structured document is based on this data fusion approach. It is called the small document approach. In this approach, each field of the document will be considered and a small document and scored against the query. At the ranking stage, these scores will be combined using data fusion techniques. For a document D consisting of k Fields F. The document score against a query Q is given by:

$$Score(Q, D) = \sum_{j=1}^{k} v_j \times Score(Q, F_j) \tag{2.22}$$

where $Score(Q, F_j)$ can be scored using any appropriate retrieval model, e.g., BM25, query likelihood language model, etc., and v_j is the weight for F_j, which is trained or assigned based on some kind of prior information. Wilkinson[39] studies combining multiple representations of document fields in retrieving structured documents comprehensively.

However, Robertson criticized this approach in that when using BM25 as the field scoring model, it can lead to poor performance, because, among other reasons, it breaks "the

carefully constructed non-linear saturation of term frequency in the BM25 function" [4]. In BM25 and most modern term weighting functions, there is a non-linearity on the term frequency component based on that "information gain on observing a term first time should be greater than the information gain on subsequently seeing the same term" (Robertson et al., 2004). The small document approach apparently will break this non-linearity, because terms may be observed multiple "first time" in a document if the fields are scored using BM25 separately. To address this issue, Robertson proposed the BM25F model, in which the original fields terms are merged into a single unstructured document and term frequencies of terms in a field are weighted based on its original field weight. For example, given a term w occurs in the title field once, and we give the title field a weight of 2, w will be merged to into the final document with a term frequency of 2 to be combined with its other occurrences. Then the final score of the document against a query is given by this new pseudo document D':

$$Score(Q, D) = Score(Q, D') = Score(Q, \sum_{j=1}^{k} v_j \ F_j) \tag{2.23}$$

where F_j is the term frequency vector for the j-th field, and v_j is its weight. This approach is called the in-model combination approach [5]. The key of this approach is to preserve the properties of the underlying retrieval model, e.g. BM25, as much as possible. Another popular in-model combination approach for semi-structured information retrieval is the mixture-based language model proposed in [40]. In this model, a language model developed for each field, the document is a mixture of the individual language models:

$$P(Q|D) = \prod_{i=1}^{m} P(q_i|D) = \prod_{i=1}^{m} \prod_{j=1}^{k} P(q_i|F_j)P(F_j|D) \tag{2.24}$$

where $P(q_i|F_j)$ is probability of generating query term q_i from field F_j, $P(F_j|D)$ can be regarded as the weight of a particular field. It is important to note that the generative nature of the model is preserved by ensuring $\sum_{j=1}^{k} P(F_j|D) = 1$. This model is found to be effective for known item search tasks[40].

In existing field-based retrieval models, both the BM25F and the mixture language model, fields are assumed to be non-repeatable, non-hierarchical and of which the terms are draw from the same vocabulary. These assumptions may not hold, when we are dealing with certain fields. For example, the author and keyword fields in literature search are usually drawn from completely different vocabularies. Therefore there is a need to address the vocabulary mismatching issue across fields when scoring the field against the query term. We have not seen any work done in this respect for semi-structured information retrieval yet.

2.4 Bibliometrics and information retrieval

Searching in scientific literature differs from that in the general domain in that the underlying scientific literature corpus is with regularities governed by the dynamics of scientific community and communications. These laws and regularities inherent in scientific literatures can be used in enhancing retrieval efficiency and effectiveness. The last century witness discovery of several important bibliometric laws, including the Bradford Laws of scattering of scientific literatures, Lotka's law of author productivity and Price's law of literature decay. These laws reveal important characteristics of the scientific literature space. But their application in information retrieval is limited.

2.4.1 Existing work on using citation in information retrieval

Recent years have seen growing interests in combining bibliometrics and information retrieval (IR), the two major specialties of information science[41]. White proposed a synthesis of the two under Sperber and Wilson's relevance theory, leading to a novel Pennant visualization for accessing literature[42]. Extensive researches have been carried on leveraging the inherent regularity and dynamics of bibliographical entities in scientific information spaces to improve search strategies and retrieval quality[43;44]. Mutschke et. al. argue that conceptualization of scholarly activity and structure in science can be used to improve retrieval quality[43]. Their examples include using co-word analysis for query expansion, using Bradford's law of information distribution pattern and co-authorship network analysis results to re-rank search results.

In practice, the idea of systematically using citation to assist searching scientific literature at least started as early as Garfield's initiation in creating citation indexes for scientific articles in the 1950s[45]. Citation following functions are integral component of existing literature search systems such as the Thomson Reuters Web of Science (formally ISI Web of Science), Citeseer[46], and Google Scholar.

The IR community also studied citation's potential in enhancing retrieval effectiveness. Salton found out that textual similarity correlated with citation similarity and proposed using terms from bibliographic citation documents to augment original document representation[47]. Many other early work on using citation relations in information retrieval is reviewed by Smith[48]. In some studies, citation is regarded as a separate/alternative representation of the textual content therefore can be used in retrieval of the document[49;50].

We can categorize recent researches on leveraging citation in retrieval as following:

- Model citation as a source of query independent score and combine them with content

based score: Yin et al. studied linearly combining content score and link score modeled under the BM25 model to improve biomedical literature retrieval[51].

- Derive language model document priors based on citation analysis: Meij and de Rijke studied deriving document priors from citation counts[52]. Zhao and Hu explored deriving document priors based on citation induced PageRank and co-citation clustering[8].

- Use citation as retrieval strategies: Larsen studies the "boomerang" effect, which is to use frequently occurring citations in top retrieval result to query against citation indexes for relevant documents[53].

- Use citation network as a relevance propagation mechanism: Norozi et al. experimented with a contextualization approach to boost document scores with the scores of their random walked neighborhood documents over the in-link and out-link citation network[54].

- Finding index terms through citation: Bradshaw proposed the idea of Reference Directed Indexing (RDI), in which document are index by terms from their citation windows[55]. Ritchie studied using citation context to enhance retrieval effectiveness[56]. They extract terms from citation index context, and use those terms to enhance the representation of the cited papers[56]. Their results shows that terms from citation context can improve the retrieval effectiveness by up to 7.4%, and weighting terms from citation context higher increase the improvement (Ritchie, 2008).

However, in some approaches, using citation brings improvements in retrieval effectiveness, while in others not. The overall question on whether citation will help improve retrieval effectiveness is inconclusive. Moreover, no study exists in using citation structure to smooth language model.

2.5 Learning to Rank for IR

Established retrieval models such as BM25 and query likelihood language model are popular because they are fast enough to be run over the entire document index and usually deliver reasonable good results. However, they mostly rely on term and document statistics features, ignoring many other potentially useful features. The process of incorporating ad hoc features into those models is cumbersome and involves a lot of tuning. For example, it is non trivial to include document independent features such as PageRank, URL length and Click Distance to the BM25 model as shown in previous research[57]. As more and more features are available, it is even more difficult to find appropriate ranking/scoring functions that can utilize them. In this respect, a new paradigm of great surging interest arises in recent years, which is to use discriminative machine learning approaches to learn implicit ranking functions[58]. In this framework, large amount of features can be easily applied, and varieties of established machine learning algorithms can be used in train rankers with training data sets.

The learning to rank problem can be formulated as given a set of query- document pairs, $< q, d >$, we want to learn a set of function $f(q, d)$ which will classify those pairs in correct order. Machine learning techniques are generally used in finding the f. There are two ways to categorize learning to rank methods, either by their loss functions or the machine learning techniques they use[59]. In the former taxonomy, learning to rank methods can be categorized into: point-wise, pair-wise and list-wise approaches. In the latter, they may be categorized into SVM-based, Boosting SVM, Neural Network-based and other approaches.

Using machine learning techniques to learn parameters for retrieval models has been explored in early 1990s[60]. But it was unsuccessful then. With the rise of the web, as more and more training data are available, machine learned techniques become more and more

used in information retrieval.

2.5.1 Point-wise Approach

In the point-wise learning to rank approach the goal is, for a input document d_ii, to learn either a real value or categorical label y_i, and then to rank all the concerned documents based on y_i. Regression and classification algorithms are generally used in point-wise approach. Representative point-wise ranking algorithms include: subset ranking[61;62], McRank[63], Prank[62], etc.

2.5.2 Pair-wise Approach

In information retrieval, we generally care more about the ordering of the returned documents than their actual scores. In other words, given we know the preferences over any pair of documents in a search result set, it is possible to construct an overall ordering of all the documents in it. With this insight, the pair-wise learning to rank approach turns the ranking problem into a binary classification problem. The input here becomes document pairs, e.g. (d_u, d_v), and the output is the preference $y_{u,v}$. Such a reformulation of the ranking problem makes it possible to use many powerful classification algorithms such as Support Vector Machines (SVM) for learning to rank tasks. Representative pair-wise ranking algorithms include Ranking SVM[64], RankBoost[65], LambdaMART[66], RankNet[67], etc.

2.5.3 List-wise Approach

Methods such as pairwise approaches has the drawback that a difference between two items on the top of the search results list has the same effect as the pair near the bottom of the list, which is counter-intuitive in a ranking setting. The list-wise approach solved this problem by learning over the whole search results list, in other words, it explicitly

consider the order effects of documents. Later researchers found that directly optimizing towards the IR metrics, e.g. $DCG@k$ has been shown an effective measures than point-wise and pair-wise approaches. The input here becomes a list of document, $\vec{d} = d_{j}{}_{j=1}^{m}$, and the output is the optimal permutation of $\vec{d}$, π_y. The difficulty in this approach is that IR metrics are not continuous, therefore are difficult to optimize. Some measures are taken to make them optimizable. Representative list-wise ranking algorithms: ListNet[68], ListMLE[69], AdaRank[70], SVMMAP[71], etc.

Overall, list-wise and pair-wise approaches generally outperform point-wise approaches. And the current state of the art learning to rank algorithms are: LambdaMART, RankBoost, RankNet and AdaRank. Many learning to rank algorithms are implemented in open source toolkits. Several existing popular learning to rank toolkits include Ranklib , jforests and sofia-ml .

2.5.4 General Process of Learning to Rank

The general process of learning to rank include sampling, learning the ranking model based on training data and application the learned model to test data. A typical learning to rank process follows a top k retrieval and feature extraction[72;73].

- Top k Retrieval: retrieve top k, e.g. 1000, documents use a state of art retrieval model, e.g. BM25, as the basis for re-ranking.
- Feature Extraction: extracting features, such as fielded weighting scheme, query features.
- Model Learning/Application: learn the ranking model based on the training data/deploy the learned model to running IR system.

Recently Dang et. al. proposed a two-stage learning to rank framework for information retrieval: first, retrieve a best subset of documents using a limited set of textual features, then train a final ranking model with a larger set of query-and document-dependent features to re-rank the subset[74]. Their experiments show that this method outperforms the general learning to rank approach, which does not optimize the top-k retrieval. This indicates that different feature groups may be appropriate for different stages of the learning to rank process.

Chapter 3: Structure-aware Retrieval Models for Literature Search

This chapter investigates structure-aware retrieval models for literature search. Mainstream IR researches are built around the unstructured document model. Documents are treated as bag of words, and document structure and entities though play great role as part of the document, are not emphasized enough in existing IR modeling approaches. This can be shown in that most reported ad hoc IR experiments are typically conducted on title and abstract parts, ignoring other information such as subject descriptors, author names, and publishing venues so on.

Academic literature information resources are generally metadata rich. For example, books are with title, author, publisher fields and papers have title, author, venue, references, citations and other fields. These fields as alternative representations can potentially help retrieval of documents. But there are not many efforts in leveraging them in building effective retrieval models. This chapter first reviews several retrieval models that can deal with semi-structured documents, and then conduct experiments with non-field, single filed and field-based retrieval models on the iSearch BK and PN sections to investigate their performance in leveraging scientific literature structure information.

3.1 Structure-aware Retrieval Models

When dealing with fields, the Principle of Combination is generally employed, which states that "effective integration of more information should lead to better IR" [49]. The justification of combining multiple representations of documents in information retrieval is that all evidence may be helpful[50]. Combination can be done in two manners: combine term

frequency level from different fields or combine scores of different fields. BM25F and PL2F belongs to the former category and Mixture of Language Models (MLM) and Probabilistic Retrieval Model for Semistructured Data (PRMS) the latter.

3.1.1 BM25F

The BM25F model is proposed for extending the BM25 model to structured information[4]. In this model, term frequencies are expanded in advance to form a joint document model, then the query are evaluated over this single document instead of the individual fields. In BM25F, the weighting scheme for a term in a fielded document is given by the following formula:

$$w = \frac{(k_1 + 1) \sum_f v_f \; tf_f}{k_1((1-b) + b\frac{dl}{avdl}) + \sum_f v_f \; tf_f} \; \log \frac{N - df + 0.5}{df + 0.5} \tag{3.1}$$

where tf_f is term frequency of a term in field f , v_f is the weight given to that field, dl is the document length, $avdl$ is the average document length, df is document frequency of the term in the whole collection, and N is the total number of documents, k_1 and b are free parameters.

3.1.2 PL2F

PL2F is another field-based weighting model from the Divergence from Randomness (DFR) framework. PL2F applied a per-field term frequency in scoring a fielded document[75]. In PL2F, the score of a document against a query is given by

$$Score(d, Q) = \sum_{t \in Q} \frac{qtf}{qtf_{max}} \frac{tfn}{tfn + 1} (tfn * log_2 \frac{tfn}{\lambda} + (\lambda - tfn) log_2 e + 0.5 \; log_2(2\pi * tfn)) \tag{3.2}$$

where qtf is the query term frequency, and qtf_{max} is the maximum query term frequency. tfn is the normalized term frequency:

$$tfn = \sum_f (w_f * tf_f * log2(1 + c_f * \frac{avgl_f}{l_f})) \tag{3.3}$$

where w_f is the parameter for the weight of field f and c_f is length normalization parameter for f.

3.1.3 Mixture of Language Models (MLM)

Mixture of language models is an approach to handle structured documents in the language modeling framework. It constructs a language model for each field, and compute the document score based on weighted combination of the individual field scores[40]. The score of query Q against D in MLM is given by:

$$P(Q|D) = \prod_{i=1}^{m} P(q_i|D) = \prod_{i=1}^{m} \prod_{j=1}^{k} P(q_i|F_j)P(F_j|D) \tag{3.4}$$

where $P(q_i|F_j)$ is probability of generating query term q_i from field F_j, $P(F_j|D)$ can be regarded as the weight of a particular field. It is important to note that the generative nature of the model is preserved by ensuing $\sum_{j=1}^{k} P(F_j|D) = 1$.

3.1.4 Probabilistic Retrieval Model for Semistructured Data (PRMS)

In aforementioned models, the weights for combining fields are often given heuristically or trained using optimization algorithms. In the probabilistic retrieval model for semistructured data, Kim proposed a method to calculate the weights based on the probability of mapping a query term to a field $P(f|q)$[76]. In this model, the probability of generating the

query Q from a fielded document D is given by:

$$P(Q|D) = \prod_{i=1}^{m} \sum_{j=1}^{n} P_M(F_j|q_i) P(q_i|F_j, D) \tag{3.5}$$

where the weights, or mapping probability of a query term q_i to a field F_j is estimated as:

$$P_M(F_j|q_i, C) = \frac{P(q_i|F, C)}{\sum_{F_k \in F} P(q_i|F_k, C)} \tag{3.6}$$

This implies that the more probable a term being generated from a field's collection language model, the larger its mapping probability to that field. To illustrate this with an example, in a semi-structured movie dataset, term "meg ryan" will have high weight to be associated with the cast field, while " romance" will have high weight to be associated with genre field.

3.2 Problem Description and Research Design

Our purpose of this study is to assess the effectiveness of aforementioned four structure-aware retrieval models for literature search. To achieve this goal, we run IR experiments of different structure-aware retrieval models on two fielded test collections. As it is nontrivial to correctly implemented all models in a single retrieval toolkit which scales to our collection, we adopt the most authoritative implementation of each model in well-known open source information retrieval libraries. Specifically, for BM25F and PL2F, we choose the Terrier toolkit[1], MLM the Indri toolkit [2] and PRMS the Galago toolkit [3] . Using different retrieval libraries necessarily make it difficult to compare the results among different structure-ware

[1] `http://terrier.org/`
[2] `http://sourceforge.net/p/lemur/`
[3] `http://sourceforge.net/p/lemur/wiki/Galago/`

retrieval models, but as our focus here is to assess effectiveness of structure-aware retrieval models for literature search instead of comparing them with each other. Therefore, in the experiments conducted with each toolkit, we compare the structure-aware model run with the non-fielded baseline run in that toolkit. The baseline is generally BM25 or Language Model with Dirichlet smoothing, the two state of the art IR baseline models.

3.3 Datasets

We use the BK and PN sections of the iSearch test collection for our experiments in this section. Table 3.1 and 3.2 show the basic statistics of the iSearch-BK and iSearch-PN sections. All collections are stemmed using the Porter stemmer and filtered out stop words with a stop word list of 741 common words from the Terrier IR toolkit.

Table 3.1: Dataset Statistics

	Number of Documents	Vocabulary Size	# of Tokens
iSearch-BK	18441	48655	573678
iSearch-PN	291244	182123	26667623

Table 3.2: iSearch BK and PN Field Statistics

	Field	# of Tokens	Avg. # of Tokens
BK	TITLE	168759	9.15
	AUTHOR	59420	3.22
	SUBJECT	89190	4.84
	DESCRIPTION	256309	13.90
PN	TITLE	2059682	7.07
	AUTHOR	1248335	4.29
	SUBJECT	1473399	5.06
	DESCRIPTION	21886207	75.15

3.4 Evaluation of BM25F and PL2F

For BM25F and PL2F models, we use the implementation in Terrier toolkit. We index the BK and PN sections of the iSearch collection using the Terrier toolkit. Both BK and PN come with four fields: TITLE, AUTHOR SUBJECT, and DESCRIPTION. For each field,

we build a separate index. We use the total 65 topics as our queries. For both BK and PN, three sets of experiments are run: (1) Non-field experiments in which all fields of a document are collapsed into a single text field and retrieval is conducted over this index using both BM25 and PL2 retrieval models. (2) Single field experiments in which separate field index are used for the retrieval. For simplification, we use only the PL2 retrieval model for these runs. (3) Field-based experiments in which field-based retrieval model BM25F and PL2F are used for the retrieval.

3.4.1 Parameter Training

For field-based retrieval model, it is challenging to find out the set of proper parameters for the model. Robertson[77] recommends several approaches for finding out parameters for the BM25F models. In this chapter, following (Macdonald, 2009)[78] we train the parameters for each field-based retrieval models. Both BM25F and PL2F have two sets of parameters, one is for per-field length normalization, and the other field weighting. We used the Simulated Annealing algorithm[79] to tune the parameters for different retrieval models.

For field based retrieval models, the learned field weights are shown in Table 3.3.

Table 3.3: Learned weights for different fields for BM25F and PL2F on iSearch- PN and iSearch-BK

	Field Weights	TITLE	AUTHOR	SUBJECT	DESCRIPTION
BK	BM25F	2.4685	22.4991	15.7894	10.9714
	PL2F	1.8645	13.6240	7.1845	5.6643
PN	BM25F	9.6350	3.9478	0.9888	5.7248
	PL2F	27.3662	2.3240	17.1356	8.5037

3.4.2 Results and Discussion

Table 3.4 and 3.5 are the experiment results for BK and PN sections respectively.

Based on the experiment results, we can see that field-based models achieve better than non field retrieval models setups for both BK and PN. The improvement in PN is greater

Table 3.4: iSearch-BK Fielded Experiment Results

	Model	map	P@10	ndcg	bpref
Non Field	PL2	0.1994	0.1407	0.3381	0.3819
	BM25	0.2025	0.1542	0.3406	0.3889
Single Field	TITLE_PL2	0.1016	0.0678	0.2334	0.3913
	AUTHOR_PL2	0.0559	0.0492	0.1711	0.2662
	SUBJECT_PL2	0.1031	0.0932	0.2383	0.3523
	DESCRIPTION_PL2	0.1174	0.1102	0.2522	0.2851
Field-based	PL2F	0.1999	0.1492	0.3433	0.4224
	BM25F	0.2096	0.1542	0.3486	0.4194

Table 3.5: iSearch-PN Fielded Experiment Results. A ▲ indicates significant improvement over BM25 baseline at the $p < 0.05$ level using two-tailed paired t-test.

	Model	map	P@10	ndcg	bpref
Non Field	PL2	0.1044	0.1359	0.2776	0.3213
	BM25	0.0946	0.1234	0.2652	0.3008
Single Field	TITLE_PL2	0.0573	0.0750	0.1680	0.2247
	AUTHOR_PL2	0.0002	0.0000	0.0072	0.0211
	SUBJECT_PL2	0.0003	0.0000	0.0104	0.0429
	DESCRIPTION_PL2	0.0997	0.1156	0.2635	0.2975
Field-based	PL2F	0.1083	0.1406	0.2868	0.3326
	BM25F	0.1231	0.1422▲	0.2919	0.3128

than that in BK. We hypothesize the quality of the metadata matters. The iSearch-BK dataset are based on the bibliographic records of the national library of Denmark while iSearch-PN is a crawl of the arXiv bibliographic records. Quality of the former should be better than that of the latter because they are created by professional librarians, while the latter are provided by the authors.

Single field runs are worse than both non-field and field-based runs. This is as expected because each field is only a part of the original document. Across the two collections, single field runs of PN is generally worse than that of BK, even though their average length is longer than that of BK. We think this again can be attributed to the metadata quality difference between the two collections.

Comparing the performance of the four fields setup, we can see that AUTHOR field is

the worst performing field in both collections. We hypothesize this is due to the vocabulary mismatch between the query and the AUTHOR field. More work needs to be done to address this vocabulary mismatch issue before AUTHOR field and other fields with similar mismatch can be used effectively in a structure-aware retrieval model.

3.5 Evaluation of PRMS

The PRMS model differs from other fielded retrieval models in that it does not predefine field weights. Instead the weight of a query term in a field is derived as a mapping probability estimated based both on field terms statistics and collection term statistics. With the implementation of PRMS in Galago search engine, we compared its performance with BM25F and baseline Dirichlet Language Model in Galago. Figure 3.1 is an example PRMS query `#prms(Diffractive optics)` converted to native query format.

We conducted the performance of PRMS and BM25F on the iSearch collection. We report the following results.

3.5.1 Results and Discussion

Table 3.6 and 3.7 gives the results of BM25F and PRMS in isearch BK and PN section respectively. In both collections, PRMS performs worse than baseline non fielded retrieval model.

Table 3.6: BK field models results. Results better than baseline are in bold. A ▲ indicates significant improvement over baseline.

	map	P@10	ndcg	bpref
baseline	0.2524	0.1915	0.3997	0.4638
prms	0.1026	0.086	0.2323	0.3395
bm25f	0.1124	0.1035	0.2345	0.3432

```
#combine:norm=false(
    #wsum:0=0.0:1=0.0:2=0.0:3=0.0(
        #dirichlet:lengths=title(
            #lengths:title:part=lengths()
            #counts:Diffractive:part=field.title())
        #dirichlet:lengths=author(
            #lengths:author:part=lengths()
            #counts:Diffractive:part=field.author())
        #dirichlet:lengths=subject(
            #lengths:subject:part=lengths()
            #counts:Diffractive:part=field.subject())
        #dirichlet:lengths=description(
            #lengths:description:part=lengths()
            #counts:Diffractive:part=field.description()))
    #wsum:0=0.2436:1=0.0462:2=0.6337:3=0.0763(
        #dirichlet:lengths=title(
            #lengths:title:part=lengths()
            #counts:optics:part=field.title())
        #dirichlet:lengths=author(
            #lengths:author:part=lengths()
            #counts:optics:part=field.author())
        #dirichlet:lengths=subject(
            #lengths:subject:part=lengths()
            #counts:optics:part=field.subject())
        #dirichlet:lengths=description(
            #lengths:description:part=lengths()
            #counts:optics:part=field.description())))
```

Figure 3.1: Example Galago PRMS Query

3.6 Evaluation of MLM

We build fielded index with the Indri search engine. Baseline: mixture model for fielded search. We used the Indri toolkit to execute a structured query on the corpus. For example, for a sample query, "manipulation nano spheres peptides immobilisation", the query syntax for mixture language model is formulated as in Figure 3.2.

3.6.1 Results and Discussion

Table 3.8 and 3.9 gives the results of MLM on iSearch BK and PN section respectively.

Table 3.7: PN field models results. Results better than baseline are in bold. A ▲ indicates significant improvement over baseline.

	map	P@10	ndcg	bpref
baseline	0.0551	0.0857	0.198	0.2431
bm25f	**0.0639**	**0.0937**	**0.1993**	0.2297
prms	0.0438	0.0762	0.1737	0.2401

```
#combine( #wsum( 3.0 manipulation.author
                 5.0 manipulation.title
                 1.0 manipulation.description
                 2.0 manipulation.subject )
          #wsum( 3.0 nano.author
                 5.0 nano.title
                 1.0 nano.description
                 2.0 nano.subject )
          #wsum( 3.0 sphere.author
                 5.0 sphere.title
                 1.0 sphere.description
                 2.0 sphere.subject )
          #wsum( 3.0 peptide.author
                 5.0 peptide.title
                 1.0 peptide.description
                 2.0 peptide.subject )
          #wsum( 3.0 immobilise.author
                 5.0 immobilise.title
                 1.0 immobilise.description
                 2.0 immobilise.subject ) )
```

Figure 3.2: Example Indri MLM Query

The MLM model performs well on the BK collection but not the PN collection.

3.7 Conclusions

In this chapter, we experiment with multiple structure-aware retrieval models on the iSearch test collection. Based on our experiments, in most case fielded retrieval models do not outperforms baseline non-field retrieval model (BM25 and language model with Dirichlet smoothing). This is in contrast with previously reported studies. We can conclude that structure-aware models though shown effective in known item finding task in earlier studies,

Table 3.8: BK fielded Indri Resutls

	map	P@10	ndcg	bpref
baseline	0.1795	0.1424	0.3233	0.3943
MLM	**0.2492**▲	**0.1847**▲	**0.3995**▲	**0.4925**▲

Table 3.9: PN fielded Indri Results.

	map	P@10	ndcg	bpref
baseline	0.1298	0.1531	0.3309	0.3585
MLM	0.1222	**0.1547**	0.3246	**0.3758**▲

do not work well in ad hoc literature retrieval task.

In Chapter 5 on learning to rank for literature search, we will further study including weighting model based features built up on fielded retrieval models and non fielded retrieval models, and query independent features into the learning to rank framework and examine their performances on enhancing retrieval effectiveness.

Chapter 4: Bibliometric-enhanced Literature Search

In this chapter, we explore using citation and co-citation analysis in enhancing literature search. We test using document priors derived from citation and co-citation analysis; we apply citation, co-citation, textual and topical induced similarity in the cluster-based retrieval framework.

4.1 Problem Definition

In this part, we are going to study using citation network to enhance literature retrieval. We hypothesize that citation information are helpful for searching literature. Our purpose is to leverage inter-document similarity derived from bibliometric analysis for literature search. We tested multiple retrieval frameworks to leverage citation in information retrieval.

We first assess the potential of using citation in literature search. An exploratory study on the distribution of relevant documents in the iSearch test collection reveals that citation based clusters show a great pattern that has the potential to be exploit for effective literature search. We then use document priors in language model based literature search. Thirdly, we explored document expansion based on neighborhood documents in terms of citation related similarity measures.

4.2 Relevant documents distribution in citation clusters

We test the cluster hypothesis in the context of citation based document expansion. Inter document similarities are based on citation.

Many efficient and scalable graph cutting algorithms can be used in partitioning the citation and co-citation graphs. The multilevel graph clustering algorithms are particular

suitable because they can handle prohibitively large graphs by eliminating the need for eigenvector computation[80]. The normalized cut version of the algorithm takes the objective function to minimize the number of edges among different partitioned subgraphs:

$$NCut(G) = \min_{V_1,\ldots,V_k} \sum_{c=1}^{k} \frac{edges(V_c, V \setminus V_c)}{degree(V_c)} \tag{4.1}$$

where $edges(A, B)$ is the sum of the edge weights between nodes in set A and set B, $edges(A, B) = \sum_{i \in A, j \in B} A_{ij}$.

4.2.1 Partition Literature Space via Cutting Citation Graphs

We first conducted a preliminary study to explore the validity of using citation network based clustering as a way to shard scientific literature for effective and efficient literature search.

4.2.2 Distribution of relevant documents in each cluster

Our results show that relevant documents are often concentrated in a few clusters resulted from graph cutting citation graphs. This indicates that it is promising to adopt a selective search strategy during the search process because the retrieval effectiveness would not be harmed.

We used the iSearch test collection in our experiment. The iSearch collection is an information retrieval test collection prepared by the iSearch team 4.1. It approximately consists of 18K book MAchine-Readable Cataloging (MARC) records (BK), 291K articles metadata (PN) and 160K PDF full text articles (PF), and 66 topics drawn from physics researchers' real information needs with corresponding relevance judgment data[7]. The iSearch collection is particularly suitable for our experiment in that it has 3.7 million extracted internal citation entries among papers in both the PN and PF sections. With these citations, we can

Table 4.1: Statistics of the Citation and Co-Citation Graph of the iSearch Collection

	Citation Graph	Co-citation Graph
Size	259,093	259,093
Volume	3,768,409	33,888,861
Clustering coeffiient	0.2261	0.5910

build both citation and co-citation graphs of the PN and PF sections. For easy to compare sharding results between the citation graph and the co-citation graph, we choose a subset of the iSearch collection, the 259,093 PN and PF documents that are cited at least once.

Two graphs are constructed, with their basic statistics reported (Table 4.1 and Figure 4.1, 4.2). We can see that the co-citation graph has a larger clustering coefficient than that of the citation graph, which is reasonable because it has more edges, thus better connected. The degree of both graphs follows a power law distribution. We then used the Graclus 4.2 graph clustering software, which implements the aforementioned normalized cut algorithm, to create a 10-cluster cut for each of the graphs. As a baseline, we also created a 10-cluster partition of the collection based on random document allocation.

For each query, we examine the number of relevant documents in each shard, and order the shards in descending order based on the number of their consisting relevant documents. For each sharding policy, we aggregated the shard ranks over all the involved topics/queries. The final results for the shard rank are plotted in Figure 4.3.

An advantageous pattern of relevant document distribution in shards for selective distributed literature search is shown from the results of our proposed sharding policy. When documents are randomly assigned to shards, they tend to be evenly distributed in different shards. However, when the shards are resulted from citation or co-citation graph cutting, a few shards consist of most of the relevant documents for the search query. These shards can then be the optimal shards for a given query. Comparing the two kinds of graphs, we

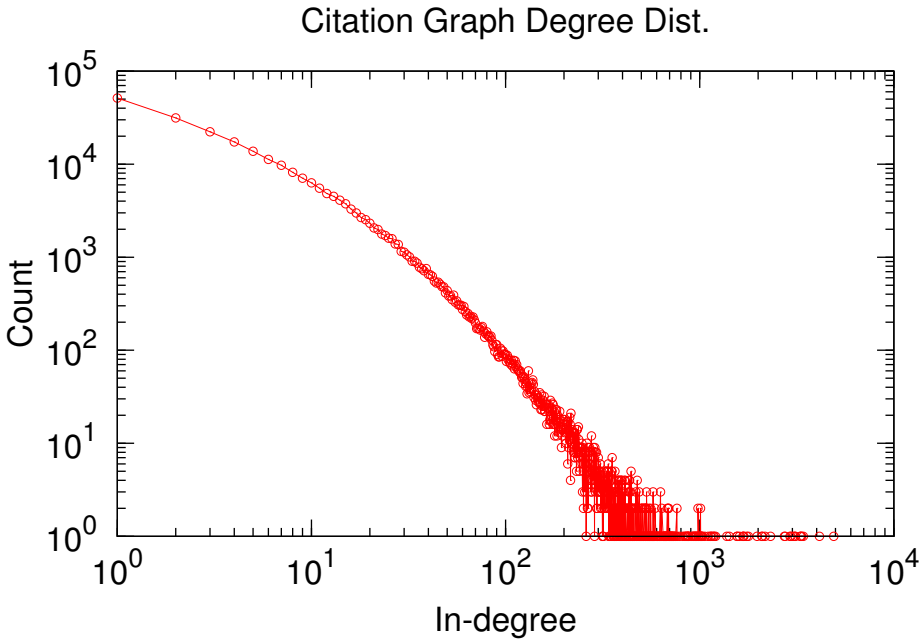

Figure 4.1: iSearch Citation Graph Degree distribution

can see that sharding based on co-citation graph partition performs even better than that of citation graph partition in terms of the potential to retrieve one or a few optimal shards. This interesting pattern indicates that sharding via citation and co-citation graph cutting is a promising direction for distributed literature search.

Our results shows that partition a scientific literature collection through citation graph partition will lead to effective search results.

4.3 Language Model Document Priors

We proposed a way to include document priors into the language model retrieval framework. We test several ways to model document citation in the language modeling for information retrieval framework.

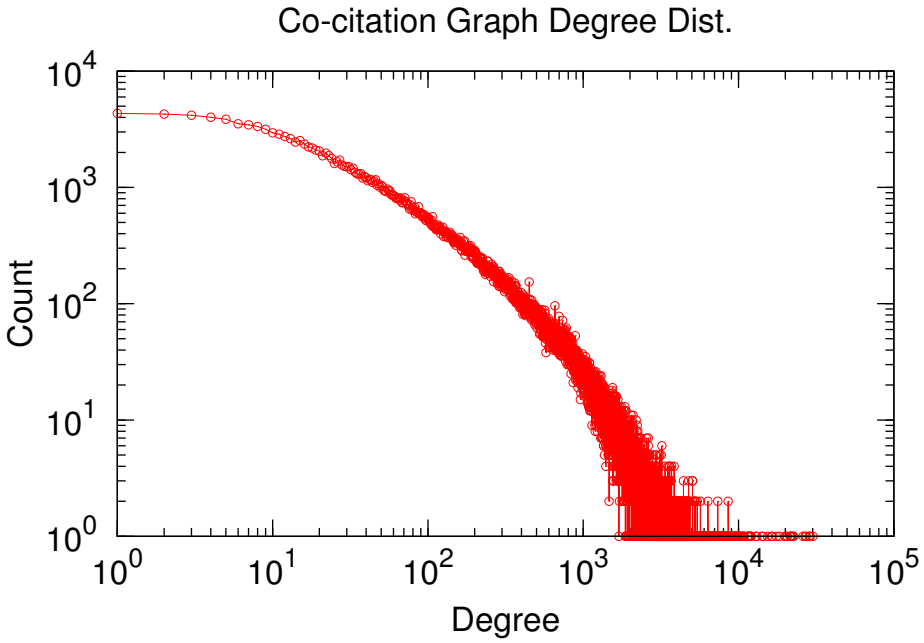

Figure 4.2: iSearch Co-citation Graph Degree distribution

4.3.1 Document Priors and Their Estimation

Analyzing paper citation and co-citation network of the iSearch dataset, we propose three kinds of document priors: paper citation count, paper PageRank score induced from citation relationships and co-citation clusters. We tested two kinds of prior estimation methods: maximum likelihood estimation (MLE) and binned estimation. For the MLE approach we also tried a logarithm version. We explain here the three kinds of document priors and how to calculate them.

Paper Citation Count Prior In this case, document prior $P(D)$ is directly estimated based on the proportion of the number of times of a paper being cited (C_i) to the total

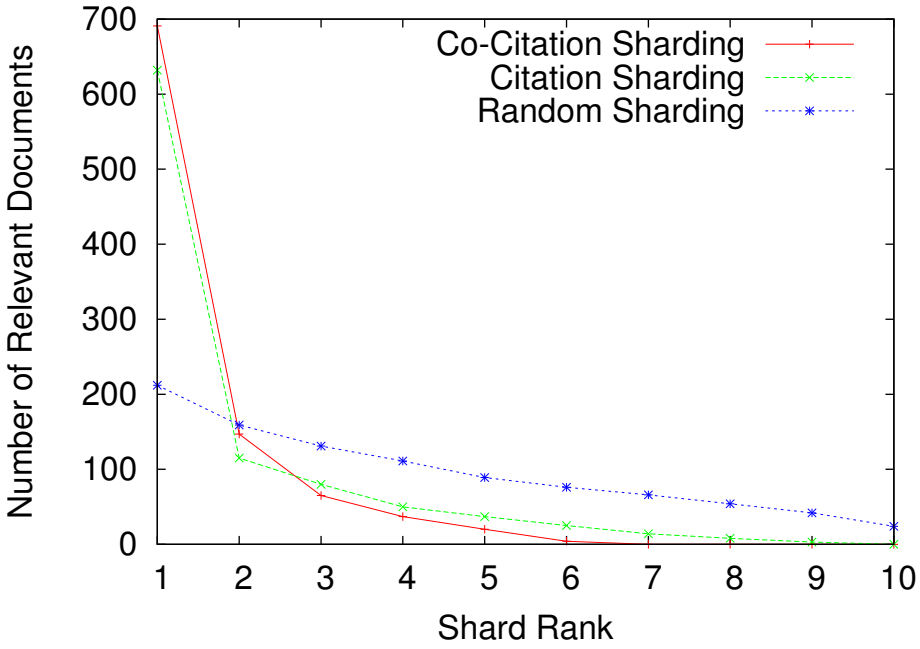

Figure 4.3: Distribution of relevant document across shards

number of times of all papers being cited:

$$P_{citedcount-mle}(D) = \frac{C_i}{\sum_{k=1}^{N} C_k}, \tag{4.2}$$

and the logarithm version:

$$P_{citedcount-\log-mle}(D) = \frac{\log(C_i)}{\sum_{k=1}^{N} \log(C_k)}. \tag{4.3}$$

Paper PageRank Prior We use the internal citation structure of the iSearch test collection to calculate the PageRank value for all the papers in our index. The PageRank value

of a given paper d is:

$$PageRank(d) = \lambda \sum_{x \in D_{*\to d}} \frac{PageRank(x)}{|D_{d\to *}|} + \frac{1-\lambda}{N}, \tag{4.4}$$

where $D_{*\to d}$ and $D_{d\to *}$ denotes papers citing d and cited by d respectively, N is the total number of papers in the collection. $\lambda = 0.85$ is called damping factor[81]. Let PR_i be the PageRank score of paper i, then document PageRank prior using MLE is:

$$P_{pagerank-mle}(D) = \frac{PR_i}{\sum_{k=1}^{N} PR_k}, \tag{4.5}$$

and the logarithm version:

$$P_{pagerank-\log-mle}(D) = \frac{\log(PR_i)}{\sum_{k=1}^{N} \log(PR_k)}. \tag{4.6}$$

Paper Co-citation Cluster Prior In this case, documents get prior probabilities based on the cluster they belong to. We calculated the document co-citation counts and compiled all the co-citation among the indexed papers, resulting a weighted undirected graph with 259,093 vertices and 33,888,861 edges, with edge weights being the number of times two papers are cited together. We then use the graph clustering software Graclus[1] to cluster the document co-citation network. Graclus provides two clustering algorithms, Normalized Cut (NCT) to minimize the sum of edge weights between clusters and Ratio Association (ASC) to maximize edge density within each clusters[80]. We tried both algorithms and decided to use NCT here because with ASC, most papers are easily clustered into one huge cluster, preventing effective prior estimation.

In the co-citation binned estimation method, the probability a document d from a given

[1] `http://www.cs.utexas.edu/users/dml/Software/graclus.html`

bin is given by:

$$P_{cocited}(D) = \frac{\#relevant\ documents\ of\ a\ bin}{\#documents\ of\ a\ bin} / \frac{\#documents\ of\ a\ bin}{\#total\ number\ of\ documents}. \quad (4.7)$$

We used a cross validation method to estimate $P(D)$ in bins. We first order the 57 topic randomly and divide them into 5 folds (11, 11, 11, 12, 12). Then at each round we use 4 folds to estimate the $P(D)$, and use the other 1 fold to test with the prior. We rotate 5 rounds, with each fold being testing set once, then we average results in all the testing folds as the final scores.

We also applied binned estimation methods on Citation Count and PageRank priors. We divide all papers into 10 bins and used the aforementioned five fold cross validation approach to geting the final scores. In total, there are 8 runs reported in Table 4.2

All estimated $P(D)$ values are converted into logarithm values and applied as Indri prior files and combined with the index using `makeprior` application of Indri. During the retrieval process, they are applied to query terms according to the Indri Query Syntax `#combine(#prior( PRIOR ) query terms)`.

4.3.2 Experiment Results and Discussion

With the baseline no prior setup, we extensively tested JelinekMercer (JM) smoothing with $\lambda \in \{0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.95, 0.99\}$, Dirichlet prior smoothing with $\mu \in \{100, 500, 800, 1000, 2000, 3000, 4000, 5000, 8000, 10000\}$, and two-stage smoothing with $\{\lambda \times \mu\}$. We find JM smoothing with $\lambda = 0.7$ performs top almost on all the four metrics we chosen. Therefore, we choose it as our retrieval model setting for the reporting baseline and other runs. For each run, we report four mainstream retrieval effec-

tiveness measurements: Mean Average Precision (MAP), Precision at 10 (P@10), NDCG and BPREF.

Table 4.2: Retrieval performance using different document priors and estimation methods compared with baseline using no prior. The best overall score is shown in bold. A ▲ indicates significant improvement over the no document prior baseline at the $p < 0.05$ level using two-tailed paired t-test.

	MAP	P@10	nDCG	BPREF
baseline-noprior	0.1152	**0.1474**	**0.3134**	**0.3079**
citedcount-mle	0.0990	0.1351	0.2825	0.2846
citedcount-log-mle	0.1092	0.1439	0.3046	0.3005
citedcount-bin10	0.1139	0.1452	0.3103	0.2943
pagerank-mle	0.1036	0.1386	0.2972	0.2941
pagerank-log-mle	0.1072	0.1421	0.3031	0.2989
pagerank-bin10	0.1137	0.1434	0.3099	0.2969
cocited-bin10	**0.1155**	0.1397	0.3122	0.3013

Table 4.2 shows our results in different setups. We can see that the overall effectiveness of applying document priors based on citation counts, PageRank and co-citation clusters comparing to our strong baseline no prior setup is limited. The only marginal improvement over the baseline happens in cocited-bin10 on MAP. Still we can still see difference across priors: overall, logarithm smoothed estimations are better than non-smoothed; binned estimations perform better than MLE estimation.

There are several possible reasons for our results. First, our relevant documents set is relatively small. The total number of relevant documents in our subset of the iSearch test collection qrels is 964, of which there are 863 distinct documents. Though that averages to 17 (964/57) relevant documents for each topic, more than half of topics (29) has only 7 or fewer documents judged as being relevant. This may contribute to the underperformance in binned estimation of document priors. Second, our current approach is totally independent to content features, only considering the citation dimension. A better approach may be to combine citation features with content features or to use document priors in a query

dependent manner. Third, performance of document priors may depend on the type of search tasks or queries.

We also conducted experiments with good query subset. By good query, we mean queries with more than 9 (inclusive) relevant documents in the collection. The results are similar to the above results. Priors are not very helpful in enhancing retrieval effectiveness.

A set of further experiments are run based on this good query set. The results are reported as in Table 4.3.

Table 4.3: Retrieval performance using different document priors and estimation methods compared with baseline using no prior on good query set. Scores better than no prior baseline are in bold. A ▲ indicates significant improvement over the no document prior baseline at the $p < 0.05$ level using two-tailed paired t-test.

	map	P@10	ndcg	bpref
good-baseline	0.1109	0.2107	0.3643	0.4178
good-CITEDCOUNT	0.0955	0.175	0.3241	0.3618
good-CITEDCOUNTBIN10	0.1102	0.1929	**0.3678**	**0.4214**
good-CITEDCOUNTLOGMLE	0.1081	0.1857	0.354	0.4041
good-COCITEDBIN10	**0.1122**	0.2107	**0.3674**	**0.4221**
good-PAGERANKBIN10	**0.1123**	0.2	**0.3662**	0.4174
good-PAGERANKLOGMLE	0.1096	0.1893	0.3563	0.402
good-PAGERANKMLE	0.1028	0.175	0.3416	0.3894

4.4 Document Expansion based on Co-Citation Analysis

Given the unpromising results on exploiting document citation and co-citation analysis results for estimating language model document priors. In this section, We explore document expansion approaches for boosting literature search. We hypothesize that by expanding documents with their most similar neighborhoods we can achieve better representation of the candidate document or boosted retrieval status value. As our focus is on bibliometric aspects, we choose to focus on similarity functions based on citation and co-citation analysis results. In terms of expansion, we test both expand with neighborhood document text, i.e. term frequencies, and with neighborhood document scores.

4.4.1 Inter-document Similarities

We choose to compare raw co-citation count, co-citation based cosine similarity, and Howard White's pennant similarity.

Co-citation Counts

This similarity is based on the raw co-citation count of two papers. Often a threshold is set to filter out low co-cited counts. We set here the threshold to 5.

Cosine Similarity based on Co-citation Counts

Given the full co-citation matrix of all the documents in the collection, the cosine similarity between two documents can be computed using Equation 4.8. We use cosine similarity to compute the inter document co-citation similarity.

$$cos(d_1, d_2) = \frac{\sum_{i=1}^{N} CC_{1i} * CC_{2i}}{\sqrt{\sum_{i=1}^{N} CC_{1i}^2}\sqrt{\sum_{i=1}^{N} CC_{2i}^2}} \tag{4.8}$$

where CC_{1i} and CC_{2_i} are documents co-cited with d_1 and d_2 respectively.

Pennant Score

Pennant score is a $tf \cdot idf$ like score proposed by Howard White as a way to depict the relationship between two descriptors[42;82]. Here we use it to describe the similarity between two co-cited papers.

The pennant score of document d_1 to document d_2 is defined as the following equation:

$$s_{<d_1,d_2>} = \log(|Cooc(d_1, d_2)| + 1) \times \log \frac{|D|}{|Cited(d_2)|} \tag{4.9}$$

where $|Cooc(d_1, d_2)|$ is the co-citation frequency of paper d_1, d_2 in the corpus, $|Cited(d_2)|$ is the total cited frequency of d_2, and $|D|$ is the total number of documents in the corpus.

It should be noted that the score is asymmetric, such that $s_{<d_1,d2>}$ is not necessary equal to $s_{<d_2,d_1>}$.

4.4.2 Document Expansion with Neighborhood Document Text

In this experiment, we expand the content of each document with its five most similar documents. In defining the similarity, we used both document co-citation count and pennant score.

Table 4.4 shows the results of the iSearch queries on these two expanded collections, comparing to the baseline run based on the original collection. The experiment is run over three indexes. The baseline consists of the full PNPFCited dataset (cf. Section 1.5.1). Run cooc5 is based on the index constructed by expanded each document with its 5 top co-cited documents, Run pennant5 is with 5 documents with top pennant scores.

Table 4.4: cooc5 and pennant5 document expansion experiment results

Run	map	P@10	ndcg	bpref
Baseline	0.0970	0.1281	0.2805	0.2808
cooc5	0.0557	0.0877	0.1951	0.2383
pennant5	0.0541	0.0825	0.2024	0.2528

We can see that performance actually downgrades when we directly expand documents with raw terms of their similar neighborhood documents. This implies that more principled ways need to be employed for the document expansion. In the following section we further this line of investigation.

4.4.3 Boosting Document Scores with Neighborhood Document Scores

We experiment several ways to boost the document score with its neighborhood documents over the citation and co-citation based similarity space. The experiments are run on the same PNPFCited collection as in previous sections.

We then use the Indri search engine to run a two step retrieval process. First do a

baseline retrieval use our previous optimized settings (cf. Section 4.3.2), then we expand this baseline document set with neighborhood documents based on the given similarity measures. Three similarity matrices are generated in advance. With the original result set and the expanded documents as the working set, we run a second retrieval run and rerank the returned documents based on Equation 4.10.

$$\hat{S(i)} = (1 - \lambda)S(i) + \lambda \sum_{j \in N(i)} S(j)w(j) \tag{4.10}$$

where the size of $N(i)$ is the major parameter we tested and $w(j)$ is normalized over all selected neighborhood documents, $S(i)$is the original score of document i and $S(j)$ is original score of neighborhood document j. $\hat{S(i)}$ is the final score used to rerank the document set. Then this final ranking is evaluated. As our underlying retrieval model is language model, the returned document scores are negative log probability values. To make the score combination working properly, we recover the original document probability by taking exponential operation, without further normalization. In the future, we will try min-max normalization on these scores before the combination.

For each of the three similarity types, we test expanding document count from 1 to 20, and tuning the interpolate parameter λ in Equation 4.10 from 0.1 to 0.9. In total, we get 630 set of results. 135 setups outperforms the baseline in terms of MAP, 41 cooccount setups and 94 pennant. Top 40 are pennant setups. The best performing setup is pennant score based expansion with 11 neighborhood documents and $\lambda = 0.8$.

Chapter 5: Learning to Rank for Literature Search

In previous chapters, we observed that there exist many sources of evidence that are potentially helpful for increasing the effectiveness of literature search. This chapter goes beyond modeling different evidences separately by investigating a consistent way to embrace all available evidences for literature search. We adopt the state of the art learning to rank (LETOR) algorithms to derive composite ranking models for literature search tasks. These machine-learned retrieval models cover a multitude of features that go beyond dominated term statistics-based features. Under this framework, we compare several LETOR algorithms as well as the performance of multiple groups of features for literature search.

Our purpose is to assess the effectiveness of structure and citation features for literature search in the LETOR framework, as well as the performance of these features and LETOR algorithms in a heterogeneous environment.

5.1 LETOR Algorithms

We employed the following learning to rank algorithms: Regression tree-based method LambdaMART[66;83] , Adarank[70], and Coordinate Ascent[84]. Adarank is a list-wise learning to rank algorithm. These algorithms are selected because of their representativeness as being the mainstream LETOR algorithms. All LETOR algorithms are based on the implementation in the Ranklib library . During the training step, we used NDCG@100 as the metric to optimize.

5.1.1 AdaRank

AdaRank is a list-wise learning to rank algorithm[70]. Given a ranking list (permutation of the retrieved documents) π, and the corresponding list of grades y, the AdaRank algorithm minimize the object function $\sum_{i=1}^{M}(1 - E(\pi_i, y_i))$, where E is a listwise evaluation measure, e.g. NDCG. Because most IR evaluation measures are generally not smooth or differentiable, direct optimization of them is difficult. Therefore AdaRank choose to optimize the upper bound of the above objective function, $\sum_{i=1}^{M} exp(-E(\pi_i, y_i))$. The learning algorithm for AdaRank is like AdaBoost in that it outputs a set of weak rankers which will be linearly combined as the final ranking model.

5.1.2 Coordinate Ascent

Coordinated Ascent is another linear feature-based ranking model[84]. It turns multivariate optimization problems into a set of single variate problems, in that each time it chooses only one parameter to optimize while holding all others fixed. This process repeats for all parameters until the objective function converges.

5.1.3 LambdaMART

LambdaMART is a boosted regression tree method. LambdaMART is MART with LambdaRank as the gradient. Instead of finding a linear combination of features, LambdaMART constructs a set of regression trees using thresholds of particular features as the decision criteria for splitting the tree.

LambdaMART has achieved good performance in several learning to rank challenges[85]. Work that proposed using bagged ensembles of LambdaMART can further improve the performance via combining boosting's low bias learning potential with bagging's lower variance potentials[86].

5.2 Problem Definition

Our problem is to leverage all the available features for learning to rank in literature search. This problem consists of the following sub questions: (1) What are the most effective feature set? (2) Which LETOR algorithms are most effective in general and can leverage structure and citation features?

5.3 Dataset

We use multiple subsets of the iSearch collection as our dataset. There are 65 queries from the iSearch test collection, of which a good portion are with fewer than 9 relevant documents. It would be difficult to reliably assess performance of learning to rank algorithms on such a small dataset. Therefore, we propose a way to expand our LETOR dataset by segmenting subsets of the queries by sessions. We choose queries that with more than 9 relevant documents into our pool; queries with too few relevant documents are detrimental as training set. We recognize that many queries are actually a set of subqueries, e.g. query 002 (see Figure 1.1) has four sub queries: nano spheres, beads, magnetic, and sorting. To expand our training and testing collections, we use these queries to generate a set of bi-subqueries as our training and testing dataset. For example, original query 002 results six new queries. The purpose of using bi-subqueries, rather than single subquery, is to balance between expanding query-document sets and preserving the meaning of the original query as much as possible for training and testing thus the confidence of the relevance label.

Table 5.1: iSearch Query Sets

	BK	PN	PF	all
Total Valid Queries	55	61	59	65
Selected Queries	16	32	30	55
Generated Bi-Subqueies	93	248	215	359

Table 5.1 gives the statistics of the original and expanded iSearch queries. With this

query set, we reuse the original qrels to prepare the learning to rank dataset.

5.4 Experiment Design

We developed a set of features for learning to rank in scientific literature search. These features can be categorized as query dependent, query independent, global or local features. Some features have strong bibliometric background, e.g. co-authorship centrality and citation related features. The goal of our experiment is to evaluate the effectiveness of different feature sets when being added to or removed from our machine learned ranking model, and how they contribute to the final ranking functions.

The same index in previous structural retrieval models experiments are used for BK and PN section. We build two additional indices, one for the PF section, PF, and one for all of the BK, PN, and PF sections, i.e. BKPNPF. We used BM25 retrieval model with query expansion to do the sampling step. Previous study shows that training parameters for weighting models is unnecessary when treating weighting model scores as features[87]. Therefore, for weighting model features, we generally used the default parameter setup, except for field-based weighting models BM25F and PL2F, of which we use the field weights trained with simulated annealing algorithms.

5.4.1 Features

Based on our review of previous work, we compile the following features for our experiments. We remarks on the details of how to compute them here.

Weighting models: BM25, P2L, LM(Dirichlet LM, Hiemstra LM, TFIDF)

Field weighting models: BM25F, PL2F and individual field weighting models.

Citation related features: cited count and paper PageRank.

Table 5.2: LETOR Features

		BK	PN	PF	all
Sampling	BM25	✓	✓	✓	✓
IR Models Features	DirichletLM	✓	✓	✓	✓
	Hiemstra_LM	✓	✓	✓	✓
	LemurTF_IDF	✓	✓	✓	✓
	TF_IDF	✓	✓	✓	✓
	DFRDependenceScore	✓	✓	✓	✓
	MRFDependenceScore		✓	✓	✓
Cite Features	Citation		✓	✓	✓
	PageRank		✓	✓	✓
Field Features	BM25F	✓	✓		✓
	PL2F	✓	✓		✓
	TITLE_BM25	✓	✓		✓
	AUTHOR_BM25	✓	✓		✓
	SUBJECT_BM25	✓	✓		✓
	DESCRIPTION_BM25	✓	✓		✓

All features valued are normalized using the following function:

$$S_s = \frac{s - s_{min}}{s_{max} - s_{min}} \tag{5.1}$$

where s is the original score, s_{min} and s_{max} are the minimal and maximum scores. For citation feature, we applied a logarithm transformation of the raw count, following previous practice[88]:

$$\hat{s} = log(1 + s) \tag{5.2}$$

where s is the original cited count, $\hat{s}$ is the transformed count.

We conduct a baseline run using the above feature sets in literature search. We use the Terrier IR toolkit to conduct our experiment. Weighting model features are generated with Terrier IR Toolkit.

We choose a five-fold cross validation setup. We first learn a ranking model with all

available features. Then an additional feature ablation process is employed to determine the contribution of different feature groups to the final outcome.

5.5 Results and Discussion

Table 5.3, 5.4, 5.5 and 5.6 report our experiment results on the four indices: BK, PN, PF and BKPNPF. The baseline ranker for all the indices is BM25. For each index, three LETOR algorithms are used: AdaRank, Coordinate Ascent and LambdaMART. The training metric is NDCG@100 and the testing metrics are MAP, NDCG@20 and NDCG@100. The results of different feature setup are reported: "all" means all applicable features are used. "fieldbased" means only sampling score and field-based weighting model features are used. "nofieldbased" means all features except field-based features are used. The same naming convention applies on "singlefield", "citation", "pagerank". Special note should be given to the "cite" which consists both "citation" and "pagerank" features and "nocite" excludes these two features. Figure 5.1, 5.2 5.3, and 5.4 show the box plots of MAP scores for different LETOR algorithm and feature group setups for all four indices.

Table 5.3: BK LETOR results. A ▲ indicates significant improvement over BM25 baseline at the $p < 0.05$ level using two-tailed paired t-test.

Ranker	FeatureSetup	MAP	NDCG@20	NDCG@100
Baseline	BM25	0.2004	0.2246	0.3192
AdaRank	all	0.2332▲	0.2436▲	0.3544▲
AdaRank	fieldbased	0.2143▲	0.2223	0.3311▲
AdaRank	nofieldbased	0.2298▲	0.2316	0.3596▲
AdaRank	nosinglefield	0.2313▲	0.2327	0.3487▲
AdaRank	singlefield	0.1979	0.2257	0.3196
CoordinateAscent	all	0.2461▲	0.2339	0.3724▲
CoordinateAscent	fieldbased	0.2283▲	0.2180	0.3369▲
CoordinateAscent	nofieldbased	0.2442▲	0.2424	0.3776▲
CoordinateAscent	nosinglefield	0.2521▲	0.2380	0.3815▲
CoordinateAscent	singlefield	0.1904	0.2180	0.3155
LambdaMART	all	0.2962▲	0.3597▲	0.4653▲
LambdaMART	fieldbased	0.2177▲	0.2257	0.3434▲

Continued...

Ranker	FeatureSetup	MAP	NDCG@20	NDCG@100
LambdaMART	nofieldbased	0.2942▲	0.3577▲	0.4638▲
LambdaMART	nosinglefield	0.2433▲	0.2580▲	0.3820▲
LambdaMART	singlefield	0.2466▲	0.3018▲	0.4089▲

Table 5.4: PN LETOR results. A ▲ indicates significant improvement over BM25 baseline at the $p < 0.05$ level using two-tailed paired t-test.

Ranker	FeatureSetup	MAP	NDCG@20	NDCG@100
Baseline	BM25	0.0976	0.1245	0.1928
AdaRank	all	0.0970	0.1150	0.1809
AdaRank	fieldbased	0.1151▲	0.1443▲	0.2113▲
AdaRank	nofieldbased	0.0954	0.1139	0.1793
AdaRank	nosinglefield	0.0844	0.0985	0.1512
AdaRank	singlefield	0.0805	0.1049	0.1757
AdaRank	cite	0.0267	0.0224	0.0444
AdaRank	nocite	0.1171▲	0.1402	0.2113▲
AdaRank	citation	0.0978	0.1265▲	0.1941
AdaRank	nocitation	0.1110	0.1354	0.2036
AdaRank	pagerank	0.0976	0.1250	0.1929
AdaRank	nopagerank	0.1111	0.1356	0.2041
CoordinateAscent	all	0.1277▲	0.1574▲	0.2319▲
CoordinateAscent	fieldbased	0.1169▲	0.1477▲	0.2133▲
CoordinateAscent	nofieldbased	0.1166▲	0.1442▲	0.2229▲
CoordinateAscent	nosinglefield	0.1239▲	0.1545▲	0.2259▲
CoordinateAscent	singlefield	0.0983	0.1254	0.1943
CoordinateAscent	cite	0.0977	0.1260	0.1939
CoordinateAscent	nocite	0.1283▲	0.1578▲	0.2340▲
CoordinateAscent	citation	0.0980	0.1264▲	0.1944
CoordinateAscent	nocitation	0.1294▲	0.1604▲	0.2356▲
CoordinateAscent	pagerank	0.0973	0.1246	0.1924
CoordinateAscent	nopagerank	0.1255▲	0.1556▲	0.2285▲
LambdaMART	all	0.1249▲	0.1548▲	0.2290▲
LambdaMART	fieldbased	0.1073▲	0.1331	0.2001
LambdaMART	nofieldbased	0.1259▲	0.1563▲	0.2295▲
LambdaMART	nosinglefield	0.1289▲	0.1589▲	0.2290▲
LambdaMART	singlefield	0.0987	0.1260	0.1937
LambdaMART	cite	0.1003	0.1295	0.2014
LambdaMART	nocite	0.1273▲	0.1555▲	0.2261▲
LambdaMART	citation	0.0965	0.1236	0.1917
LambdaMART	nocitation	0.1329▲	0.1621▲	0.2336▲
LambdaMART	pagerank	0.0972	0.1196	0.1890
LambdaMART	nopagerank	0.1233▲	0.1535▲	0.2268▲

Table 5.5: PF LETOR results. A ▲ indicates significant improvement over BM25 baseline at the $p < 0.05$ level using two-tailed paired t-test.

Ranker	FeatureSetup	MAP	NDCG@20	NDCG@100
Baseline	BM25	0.1176	0.1608	0.2291
AdaRank	all	0.0502	0.0614	0.1084
AdaRank	cite	0.0762	0.1015	0.1535
AdaRank	nocite	0.0746	0.1058	0.1671
AdaRank	citation	0.1169	0.1619	0.2292
AdaRank	nocitation	0.0594	0.0800	0.1330
AdaRank	pagerank	0.1176	0.1618	0.2288
AdaRank	nopagerank	0.0570	0.0792	0.1294
CoordinateAscent	all	0.1179	0.1791	0.2507▲
CoordinateAscent	cite	0.1130	0.1554	0.2244
CoordinateAscent	nocite	0.1258	0.1874▲	0.2613▲
CoordinateAscent	citation	0.1174	0.1617	0.2284
CoordinateAscent	nocitation	0.1205	0.1806	0.2546▲
CoordinateAscent	pagerank	0.1163	0.1612	0.2277
CoordinateAscent	nopagerank	0.1220	0.1872▲	0.2572▲
LambdaMART	all	0.1257	0.1874▲	0.2573▲
LambdaMART	cite	0.1117	0.1560	0.2257
LambdaMART	nocite	0.1272	0.1833	0.2535▲
LambdaMART	citation	0.1126	0.1587	0.2256
LambdaMART	nocitation	0.1260	0.1881▲	0.2565▲
LambdaMART	pagerank	0.1097	0.1539	0.2219
LambdaMART	nopagerank	0.1288	0.1894▲	0.2588▲

Table 5.6: BKPNPF LETOR results. A ▲ indicates significant improvement over BM25 baseline at the $p < 0.05$ level using two-tailed paired t-test.

Ranker	FeatureSetup	MAP	NDCG@20	NDCG@100
Baseline	BM25	0.1107	0.1297	0.1966
AdaRank	all	0.1161	0.1366	0.2052▲
AdaRank	fieldbased	0.0256	0.0153	0.0286
AdaRank	nofieldbased	0.1161	0.1366	0.2052▲
AdaRank	nosinglefield	0.1097	0.1313	0.2030
AdaRank	singlefield	0.1127	0.1344	0.2030▲
AdaRank	cite	0.1179▲	0.1385▲	0.2050▲
AdaRank	nocite	0.1161	0.1366	0.2052▲
AdaRank	citation	0.1167▲	0.1371▲	0.2050▲
AdaRank	nocitation	0.1161	0.1366	0.2052▲
AdaRank	pagerank	0.1105	0.1272	0.1938
AdaRank	nopagerank	0.1166	0.1376	0.2074▲
CoordinateAscent	all	0.1217▲	0.1447▲	0.2157▲

Continued...

Ranker	FeatureSetup	MAP	NDCG@20	NDCG@100
CoordinateAscent	fieldbased	0.1114	0.1311	0.2022▲
CoordinateAscent	nofieldbased	0.1191▲	0.1410▲	0.2153▲
CoordinateAscent	nosinglefield	0.1176▲	0.1389▲	0.2117▲
CoordinateAscent	singlefield	0.1150	0.1383	0.2064▲
CoordinateAscent	cite	0.1142	0.1350	0.2029▲
CoordinateAscent	nocite	0.1180▲	0.1388▲	0.2103▲
CoordinateAscent	citation	0.1179▲	0.1380▲	0.2055▲
CoordinateAscent	nocitation	0.1182▲	0.1391▲	0.2124▲
CoordinateAscent	pagerank	0.1070	0.1263	0.1932
CoordinateAscent	nopagerank	0.1193▲	0.1418▲	0.2138▲
LambdaMART	all	0.1230▲	0.1512▲	0.2240▲
LambdaMART	fieldbased	0.1106	0.1293	0.2009
LambdaMART	nofieldbased	0.1278▲	0.1580▲	0.2246▲
LambdaMART	nosinglefield	0.1236▲	0.1522▲	0.2180▲
LambdaMART	singlefield	0.1123	0.1361	0.2036
LambdaMART	cite	0.1184▲	0.1426▲	0.2065▲
LambdaMART	nocite	0.1191	0.1486▲	0.2208▲
LambdaMART	citation	0.1163	0.1354	0.2025
LambdaMART	nocitation	0.1189	0.1493▲	0.2202▲
LambdaMART	pagerank	0.1092	0.1311	0.1986
LambdaMART	nopagerank	0.1235▲	0.1518▲	0.2229▲

The overall performance of learning to rank for literature search is better than the baseline retrieval models. Adding features to the baseline model will enhance the retrieval performance.

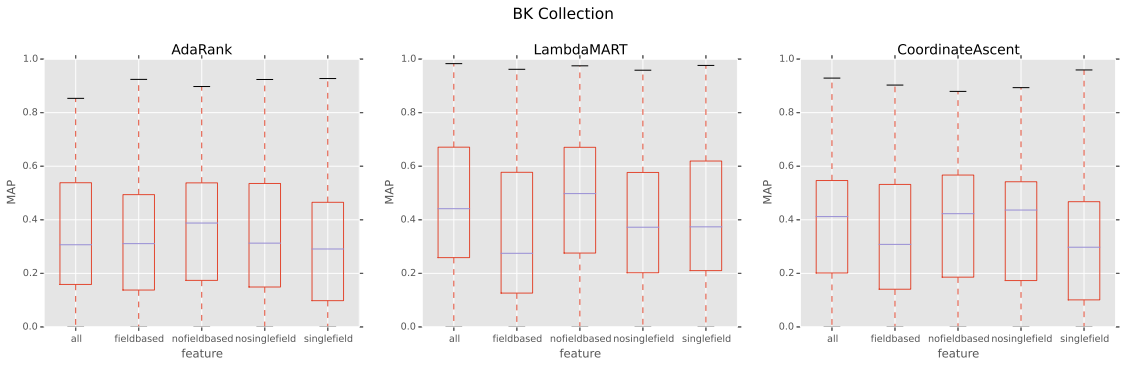

Figure 5.1: Performance of different algorithms on BK collection

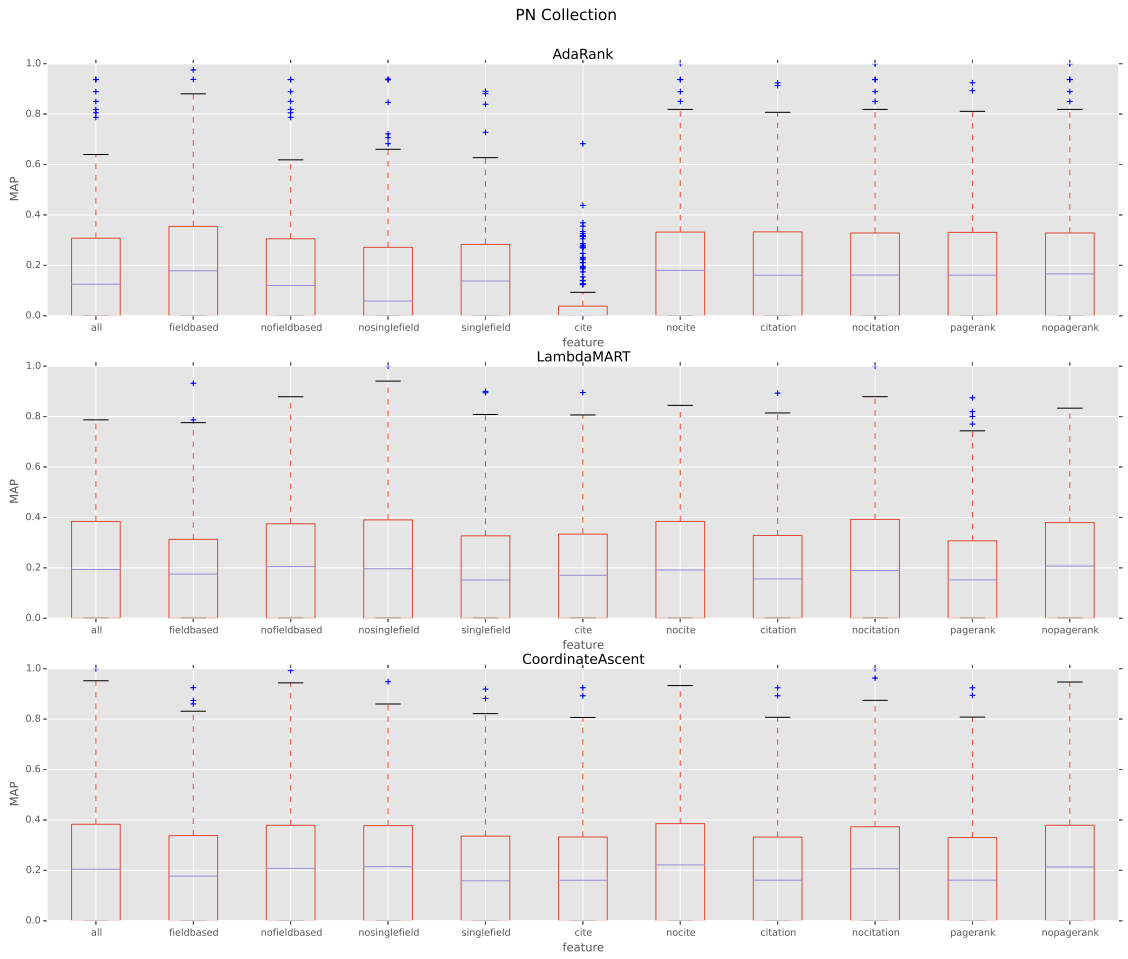

Figure 5.2: Performance of different algorithms on PN collection

5.5.1 Comparison of LETOR algorithms

According to NDCG@100, with all features used, LambdaMART is generally the best performing learning algorithm, except in PN, of which is Coordinate Ascent.

Best overall setup

According to NDCG@100, the best overall setup in BK is LambdaMART with all features, PN Coordinate Ascent with "nocite" feature setup, PF LambdaMART with "nopagerank" setup, BKPNPF LambdaMART with "nofieldbased" feature setup.

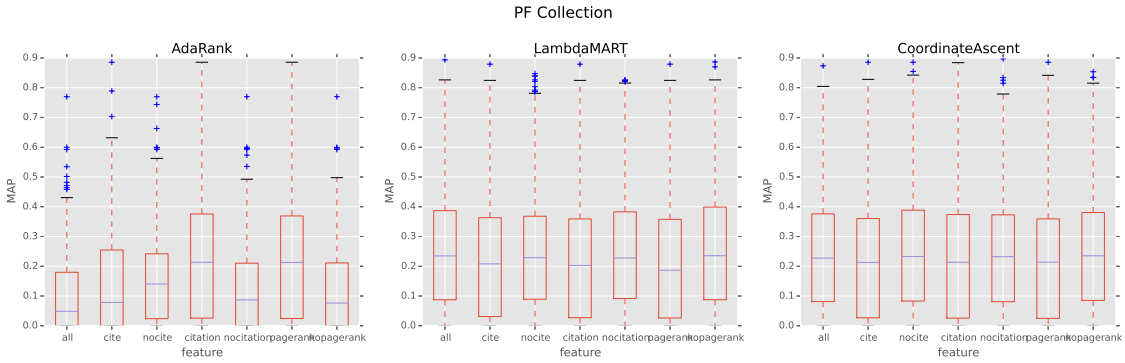

Figure 5.3: Performance of different algorithms on PF collection

5.5.2 Comparison of Features

Field features

Field features consist of PL2F, BM25F and single field BM25. The concerned indices are BK, PN and BKPNPF. Selecting LambdaMART as the LETOR algorithm and NDCG@100 as the evaluation metric, we can examine the contribution of field related features in the concerned indices.

For BK: With "fieldbased" features, NDCG@100 increases from 0.3192 to 0.34686 (+8%). without "fieldbased" features, NDCG@100 drops from 0.46984 to 0.46834 (-0.3%). This indicates that even though "fieldbased" features are helpful for enhance the LETOR algorithms, its contribution is mitigated by other weighting model features. With "singlefield" features, NDCG@100 increases from 0.3192 to 0.41204 (+29%). without "singlefield" features, NDCG@100 drops from 0.46984 to 0.38548 (-17%). This indicates single field features are much more important than "fieldbased" features in learning to rank for the BK section.

For PN: With "fieldbased" features, NDCG@100 increases from 0.1928 to 0.2001 (+3%). without "fieldbased" features, NDCG@100 increase from 0.2291 to 0.2296 (+0.2%). With

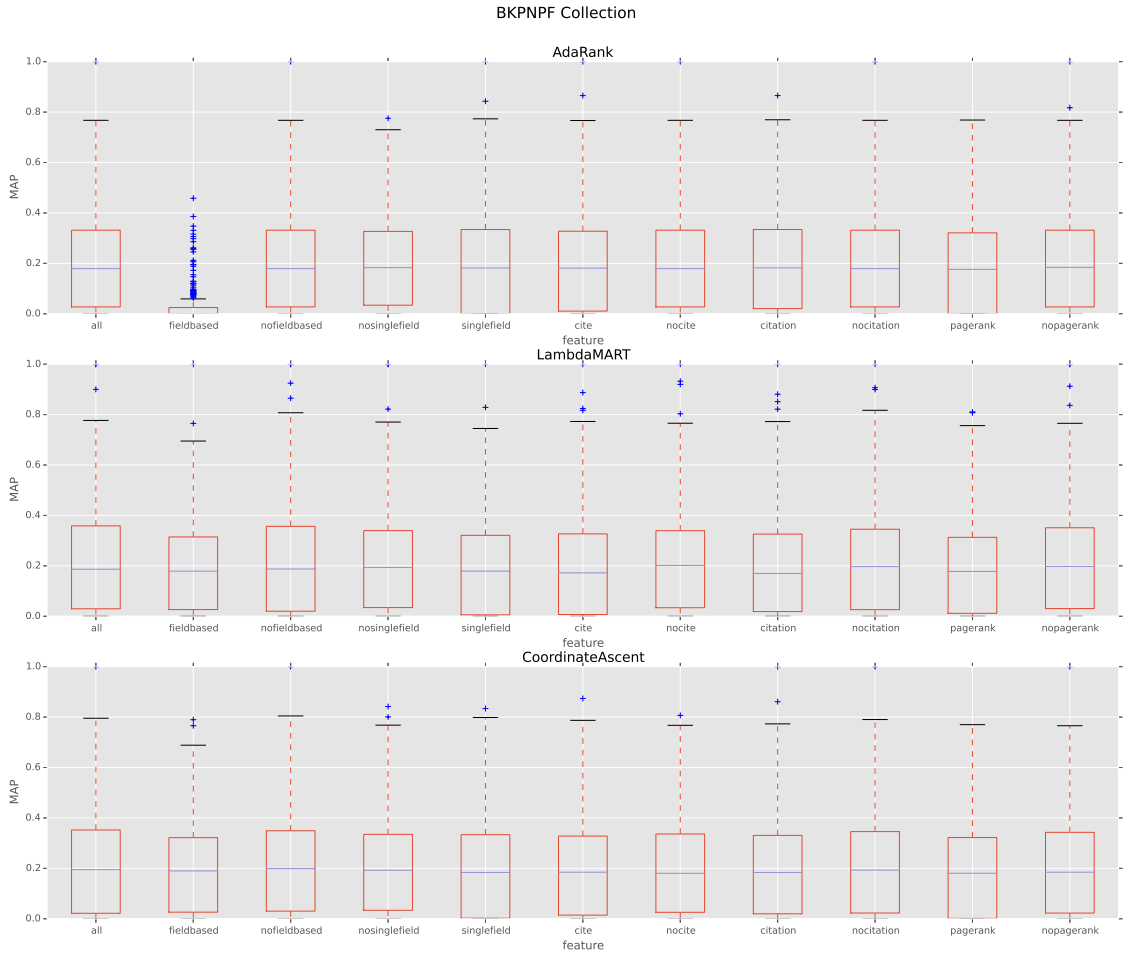

Figure 5.4: Performance of different algorithms on BKPNPF collection

"singlefield" features, NDCG@100 increases from 0.1928 to 0.1938 (+0.5%). without "singlefield" features, NDCG@100 drops from 0.2291 to 0.2290 (-0.04%). Both indicate effect of field related features is almost negligible for learning to rank for PN.

For BKPNPF: With "fieldbased" features, NDCG@100 increases from 0.1966 to 0.2010 (+2%). without "fieldbased" features, NDCG@100 increase from 0.2241 to 0.2247 (+0.2%). With "singlefield" features, NDCG@100 increases from 0.1966 to 0.2037 (+3%). without "singlefield" features, NDCG@100 drops from 0.2241 to 0.2181 (-2%). These indicate the effect of field related features in a heterogeneous environment is minor.

Cite features

Cite features consist of citation and pagerank. Collections that involves citation related features are PN, PF and BKPNPF. Selecting LambdaMART as the LETOR algorithm and NDCG@100 as the evaluation metric, we can examine the contribution of field related features in the concerned indices.

For PN: With "cite" features, NDCG@100 increases from 0.1928 to 0.2013 (+4%). without "cite" features, NDCG@100 drops from 0.2291 to 0.2263 (-1%).

For PF: With "cite" features, NDCG@100 drops from 0.2291 to 0.2257 (-1%). without "cite" features, NDCG@100 drops from 0.2547 to 0.2535 (-0.4%).

For BKPNPF: With "cite" features, NDCG@100 increases from 0.1966 to 0.2067 (+5%). without "cite" features, NDCG@100 drops from 0.2241 to 0.2208 (-1%).

The above results indicate citation related features contribute mildly for learning to rank for literature search in the iSearch test collection. This may attribute to the low quality of the citation data.

5.6 Conclusions and Future Work

The overall effectiveness of learning to rank techniques for scientific literature search is good; with more weighting models features added, the performance gets better than the BM25 baseline. Our detailed analysis of field and citation related features indicates that only in BK do field related features has a strong contribution, in other settings, the effect of these two category of features is little.

In this work, we used BM25 as the sampling algorithm. It is possible to try other algorithms, e.g. field-based retrieval models, to do sampling and to see whether there is any improvement. Also other evaluation metrics, such as $ERR@k$, instead of $NDCG@100$, can

be used during training step.

The current research focuses on common retrieval model and structured retrieval features and citation related features. In the future, we will study more bibliometric-entity related features, such as author and venue features.

Chapter 6: Conclusions

We conclude this dissertation with the following findings:

- Current structure-aware retrieval models are not effective for ad hoc scientific literature retrieval task.
- Cluster scientific literature based on citation and co-citation graph cutting is promising for implementing selective search strategies for search scientific literature.
- Under the learning to rank framework, field and citation related features are only modest helpful when other weighting model features are used.

6.1 Future Work

There are several directions can be furthered based on this dissertation.

6.1.1 Structure and Annotation enhanced Search

With the progress in natural language processing and automatic information extraction and annotation studies, there shall be more fielded information resource. Semi-structured information and entity annotated information resources are expected to explode. Structure-aware retrieval models will be useful for retrieval with these annotations.

6.1.2 Literature Search

Literature search becomes a fruitful research domain as more and more scientific literature information goes to open access. Our research can be furthered with better corpus of scientific literature.

Bibliography

[1] Peter Ingwersen, Marianne Lykke, Toine Bogers, Birger Larsen, and Haakon Lund. Assessors' search result satisfaction associated with relevance in a scientific domain. In *Proceedings of the Third Symposium on Information Interaction in Context*, IIiX '10, page 283–288, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0247-0. doi: 10.1145/1840784.1840826. URL http://doi.acm.org/10.1145/1840784.1840826.

[2] Lawrence Hunter and K. Bretonnel Cohen. Biomedical language processing: Perspective what's beyond PubMed? *Molecular cell*, 21(5):589–594, March 2006. ISSN 1097-2765. doi: 10.1016/j.molcel.2006.02.012. URL http://www.ncbi.nlm.nih.gov/pmc/articles/PMC1702322/.

[3] Gary Marchionini, S. Dwiggins, A. Katz, and Xia Lin. Information seeking in full-text end-user-oriented search systems: The roles of domain and search expertise. *Library and Information Science Research*, 15(1):35–69, 1993.

[4] Stephen E. Robertson, Hugo Zaragoza, and Michael Taylor. Simple BM25 extension to multiple weighted fields. In *Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management*, CIKM '04, page 42–49, New York, NY, USA, 2004. ACM. ISBN 1-58113-874-1. doi: 10.1145/1031171.1031181. URL http://doi.acm.org/10.1145/1031171.1031181.

[5] Paul Ogilvie. *Retrieval using Document Structure and Annotations*. Dissertation, Carnegie Mellon University, 2010.

[6] Diana Ransgaard Sørensen, Toine Bogers, and Birger Larsen. An exploration of retrieval-enhancing methods for integrated search in a digital library. In *Proceedings of the ECIR 2012 Workshop on Task-Based and Aggregated Search (TBAS2012)*, page 4–8, 2012.

[7] Marianne Lykke, Birger Larsen, Haakon Lund, and Peter Ingwersen. Developing a test collection for the evaluation of integrated search. In Cathal Gurrin, Yulan He, Gabriella Kazai, Udo Kruschwitz, Suzanne Little, Thomas Roelleke, Stefan Rüger, and Keith van Rijsbergen, editors, *Advances in Information Retrieval ECIR 2010*, number 5993 in Lecture Notes in Computer Science, pages 627–630. Springer Berlin Heidelberg, January 2010. ISBN 978-3-642-12274-3, 978-3-642-12275-0. URL http://link.springer.com/chapter/10.1007/978-3-642-12275-0_63.

[8] Haozhen Zhao and Xiaohua Hu. Language model document priors based on citation and co-citation analysis. In *BIR 2014: ECIR Workshop on Bibliometric-enhanced Information Retrieval*, 2014. URL http://ceur-ws.org/Vol-1143/paper4.pdf.

[9] Chris Buckley and Ellen M. Voorhees. Retrieval evaluation with incomplete information. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '04, page 25–32, New York, NY, USA, 2004. ACM. ISBN 1-58113-881-4. doi: 10.1145/1008992.1009000. URL http://doi.acm.org/10.1145/1008992.1009000.

[10] Gerard Salton. *Automatic Information Organization and Retrieval*. McGraw-Hill, New York, 1968.

[11] Peter Ingwersen and Kalervo Järvelin. *The Turn: Integration of Information Seeking and Retrieval in Context*. Springer, 2005.

[12] Karen Spärck Jones, S. Walker, and Stephen E. Robertson. A probabilistic model of information retrieval: development and comparative experiments: Part 1. *Information Processing & Management*, 36(6):779–808, November 2000. ISSN 0306-4573. doi: 10.1016/S0306-4573(00)00015-7. URL `http://www.sciencedirect.com/science/article/pii/S0306457300000157`.

[13] Karen Spärck Jones, S. Walker, and Stephen E. Robertson. A probabilistic model of information retrieval: development and comparative experiments: Part 2. *Information Processing & Management*, 36(6):809–840, November 2000. ISSN 0306-4573. doi: 10.1016/S0306-4573(00)00016-9. URL `http://www.sciencedirect.com/science/article/pii/S0306457300000169`.

[14] Stephen E. Robertson. The probability ranking principle in IR. *Journal of Documentation*, 33(4):294–304, 1977.

[15] W. Bruce Croft and John D. Lafferty. *Language Modeling for Information Retrieval*, volume 13 of *The Information Retrieval Series*. Kluwer Academic Publishers, Norwell, MA, USA, 2003. URL `http://www.springer.com/computer/database+management+%26+information+retrieval/book/978-1-4020-1216-7`.

[16] ChengXiang Zhai. Statistical language models for information retrieval. *Synthesis Lectures on Human Language Technologies*, 1(1):1–141, 2009.

[17] Jay M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, page 275–281, New York, NY, USA, 1998. ACM. ISBN 1-58113-015-5. doi: 10.1145/290941.291008. URL `http://doi.acm.org/10.1145/290941.291008`.

[18] Wessel Kraaij, Thijs Westerveld, and Djoerd Hiemstra. The importance of prior probabilities for entry page search. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '02, page 27–34, New York, NY, USA, 2002. ACM. ISBN 1-58113-561-0. doi: 10.1145/564376.564383. URL `http://doi.acm.org/10.1145/564376.564383`.

[19] Yun Zhou and W. Bruce Croft. Document quality models for web ad hoc retrieval. In *CIKM '05 Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 331–332, Bremen, Germany, 2005. ACM. ISBN 1-59593-140-6. doi: 10.1145/1099554.1099652. URL `http://portal.acm.org/citation.cfm?id=1099652`.

[20] Xiaoyan Li and W. Bruce Croft. Time-based language models. In *Proceedings of the Twelfth International Conference on Information and Knowledge Management*, CIKM '03, page 469–475, New York, NY, USA, 2003. ACM. ISBN 1-58113-723-0. doi: 10.1145/956863.956951. URL `http://doi.acm.org/10.1145/956863.956951`.

[21] ChengXiang Zhai and John D. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22(2):179–214, April 2004. ISSN 1046-8188. doi: 10.1145/984321.984322. URL http://doi.acm.org/10.1145/984321.984322.

[22] Adam Berger and John D. Lafferty. Information retrieval as statistical translation. In *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '99, page 222–229, New York, NY, USA, 1999. ACM. ISBN 1-58113-096-1. doi: 10.1145/312624.312681. URL http://doi.acm.org/10.1145/312624.312681.

[23] Xiaohua Zhou, Xiaodan Zhang, and Xiaohua Hu. Semantic smoothing of document models for agglomerative clustering. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, IJCAI'07, page 2922–2927, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc. URL http://dl.acm.org/citation.cfm?id=1625275.1625746.

[24] Xiaohua Zhou, Xiaohua Hu, Xiaodan Zhang, Xia Lin, and Il-Yeol Song. Context-sensitive semantic smoothing for the language modeling approach to genomic IR. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '06, page 170–177, New York, NY, USA, 2006. ACM. ISBN 1-59593-369-7. doi: 10.1145/1148170.1148203. URL http://doi.acm.org/10.1145/1148170.1148203.

[25] Xiaohua Zhou, Xiaohua Hu, and Xiaodan Zhang. Topic signature language models for ad hoc retrieval. *IEEE Transactions on Knowledge and Data Engineering*, 19(9):1276–1287, 2007. ISSN 1041-4347. doi: 10.1109/TKDE.2007.1058.

[26] John D. Lafferty and ChengXiang Zhai. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, page 111–119, New York, NY, USA, 2001. ACM. ISBN 1-58113-331-6. doi: 10.1145/383952.383970. URL http://doi.acm.org/10.1145/383952.383970.

[27] Kenneth P. Burnham and David R. Anderson. *Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach*. Springer, July 2002. ISBN 9780387953649. URL http://books.google.co.uk/books?id=fT1Iu-h6E-oC.

[28] Victor Lavrenko and W. Bruce Croft. Relevance based language models. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, page 120–127, New York, NY, USA, 2001. ACM. ISBN 1-58113-331-6. doi: 10.1145/383952.383972. URL http://doi.acm.org/10.1145/383952.383972.

[29] Nick Jardine and C. J. van Rijsbergen. The use of hierarchic clustering in information retrieval. *Information Storage and Retrieval*, 7(5):217–240, December 1971. ISSN 0020-0271. doi: 10.1016/0020-0271(71)90051-9. URL http://www.sciencedirect.com/science/article/pii/0020027171900519.

[30] Xiaoyong Liu. *Cluster-based retrieval from a language modeling perspective*. PhD dissertation, University of Massachusetts Amherst, United States – Massachusetts, 2008. URL http://search.proquest.com/docview/304565967/abstract?accountid=10559.

[31] Oren Kurland and Lillian Lee. Corpus structure, language models, and ad hoc information retrieval. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '04, page 194–201, New York, NY, USA, 2004. ACM. ISBN 1-58113-881-4. doi: 10.1145/1008992.1009027. URL http://doi.acm.org/10.1145/1008992.1009027.

[32] Tao Tao and ChengXiang Zhai. Regularized estimation of mixture models for robust pseudo-relevance feedback. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '06, page 162–169, New York, NY, USA, 2006. ACM. ISBN 1-59593-369-7. doi: 10.1145/1148170.1148201. URL http://doi.acm.org/10.1145/1148170.1148201.

[33] Xiaoyong Liu and W. Bruce Croft. Cluster-based retrieval using language models. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '04, page 186–193, New York, NY, USA, 2004. ACM. ISBN 1-58113-881-4. doi: 10.1145/1008992.1009026. URL http://doi.acm.org/10.1145/1008992.1009026.

[34] Xiaoyong Liu and W. Bruce Croft. Representing clusters for retrieval. In *SIGIR '06 Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 671–672, Seattle, Washington, USA, 2006. ACM. ISBN 1-59593-369-7. doi: 10.1145/1148170.1148310.

[35] Douglass R. Cutting, David R. Karger, Jan O. Pedersen, and John W. Tukey. Scatter/Gather: a cluster-based approach to browsing large document collections. In *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '92, page 318–329, New York, NY, USA, 1992. ACM. ISBN 0-89791-523-2. doi: 10.1145/133160.133214. URL http://doi.acm.org/10.1145/133160.133214.

[36] Xiaodan Zhang, Xiaohua Hu, and Xiaohua Zhou. A comparative evaluation of different link types on enhancing document clustering. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, page 555–562, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-164-4. doi: 10.1145/1390334.1390429.

[37] Edward A. Fox and Joseph A. Shaw. Combination of multiple searches. In *Proceedings of the 2nd Text REtrieval Conference (TREC-2) NIST SPECIAL PUBLICATION SP*, page 243–243, 1994.

[38] Mark Montague and Javed A. Aslam. Condorcet fusion for improved retrieval. In *Proceedings of the Eleventh International Conference on Information and Knowledge Management*, CIKM '02, page 538–548, New York, NY, USA, 2002. ACM. ISBN 1-58113-492-4. doi: 10.1145/584792.584881. URL http://doi.acm.org/10.1145/584792.584881.

[39] Ross Wilkinson. Effective retrieval of structured documents. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '94, page 311–317, New York, NY, USA, 1994. Springer-Verlag New York, Inc. ISBN 0-387-19889-X. URL `http://dl.acm.org/citation.cfm?id=188490.188591`.

[40] Paul Ogilvie and Jamie Callan. Combining document representations for known-item search. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, SIGIR '03, page 143–150, New York, NY, USA, 2003. ACM. ISBN 1-58113-646-3. doi: 10.1145/860435.860463. URL `http://doi.acm.org/10.1145/860435.860463`.

[41] Howard D. White and Katherine W. McCain. Visualizing a discipline: An author co-citation analysis of information science, 1972-1995. *Journal of the American Society for Information Science*, 49(4):327–355, 1998.

[42] Howard D. White. Combining bibliometrics, information retrieval, and relevance theory, part 1: First examples of a synthesis. *Journal of the American Society for Information Science and Technology*, 58(4):536–559, 2007. URL `http://onlinelibrary.wiley.com/doi/10.1002/asi.20543/full`.

[43] Peter Mutschke, Philipp Mayr, Philipp Schaer, and York Sure. Science models as value-added services for scholarly information systems. *Scientometrics*, 89(1): 349–364, 2011. URL `http://dblp.uni-trier.de/db/journals/scientometrics/scientometrics89.html#MutschkeMSS11`.

[44] Philipp Mayr and Peter Mutschke. Bibliometric-enhanced retrieval models for big scholarly information systems. In *IEEE International Conference on Big Data (IEEE BigData 2013). Workshop on Scholarly Big Data: Challenges and Ideas*, 2013. doi: 10.1109/BigData.2013.6691762.

[45] Eugene Garfield. Citation indexes for science. *Science*, 122:108–111, 1955.

[46] C. Lee Giles, Kurt D. Bollacker, and Steve Lawrence. CiteSeer: an automatic citation indexing system. In *Proceedings of the third ACM conference on Digital libraries*, page 89–98, 1998.

[47] Gerard Salton. Associative document retrieval techniques using bibliographic information. *J. ACM*, 10(4):440–457, October 1963. ISSN 0004-5411. doi: 10.1145/321186.321188. URL `http://doi.acm.org/10.1145/321186.321188`.

[48] Linda C. Smith. Citation analysis. *Library Trends*, 30(1):83–106, 1981.

[49] Edward A. Fox, G. L. Nunn, and W. C. Lee. Coefficients of combining concept classes in a collection. In *Proceedings of the 11th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '88, page 291–307, New York, NY, USA, 1988. ACM. ISBN 2-7061-0309-4. doi: 10.1145/62437.62465. URL `http://doi.acm.org/10.1145/62437.62465`.

[50] W. Bruce Croft. Combining approaches to information retrieval. In W. Bruce Croft, editor, *Advances in Information Retrieval ECIR 2000*, number 7 in The Information Retrieval Series, pages 1–36. Springer US, January 2000. ISBN 978-0-7923-7812-9, 978-0-

306-47019-6. URL http://link.springer.com/chapter/10.1007/0-306-47019-5_1.

[51] Xiaoshi Yin, Jimmy Xiangji Huang, and Zhoujun Li. Mining and modeling linkage information from citation context for improving biomedical literature retrieval. *Information Processing & Management*, 47(1):53–67, January 2011. ISSN 0306-4573. doi: 10.1016/j.ipm.2010.03.010. URL http://www.sciencedirect.com/science/article/pii/S0306457310000300.

[52] Edgar Meij and Maarten de Rijke. Using prior information derived from citations in literature search. In *RIAO '07 Large Scale Semantic Access to Content (Text, Image, Video, and Sound)*, RIAO '07, page 665–670, Paris, France, France, 2007. URL http://dl.acm.org/citation.cfm?id=1931390.1931454.

[53] Birger Larsen. *References and citations in automatic indexing and retrieval systems : experiments with the boomerang effect*. PhD dissertation, Royal School of Library and Information Science, 2004.

[54] Muhammad Ali Norozi, Arjen P. de Vries, and Paavo Arvola. Contextualization from the bibliographic structure. In *Proceedings of the ECIR 2012 Workshop on Task-Based and Aggregated Search (TBAS2012)*, page 9, 2012.

[55] Shannon Glenn Bradshaw. Reference directed indexing: Redeeming relevance for subject search in citation indexes. In Traugott Koch and Ingeborg Torvik Sølvberg, editors, *Research and Advanced Technology for Digital Libraries*, number 2769 in Lecture Notes in Computer Science, pages 499–510. Springer Berlin Heidelberg, January 2003. ISBN 978-3-540-40726-3, 978-3-540-45175-4. URL http://link.springer.com/chapter/10.1007/978-3-540-45175-4_45.

[56] Anna Ritchie, Simone Teufel, and Stephen E. Robertson. Using terms from citations for IR: some first results. In Craig Macdonald, Iadh Ounis, Vassilis Plachouras, Ian Ruthven, and Ryen W. White, editors, *Advances in Information Retrieval ECIR 2008*, number 4956 in Lecture Notes in Computer Science, pages 211–221. Springer Berlin Heidelberg, January 2008. ISBN 978-3-540-78645-0, 978-3-540-78646-7. URL http://link.springer.com/chapter/10.1007/978-3-540-78646-7_21.

[57] Nick Craswell, Stephen E. Robertson, Hugo Zaragoza, and Michael Taylor. Relevance weighting for query independent evidence. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '05, page 416–423, New York, NY, USA, 2005. ACM. ISBN 1-59593-034-5. doi: 10.1145/1076034.1076106. URL http://doi.acm.org/10.1145/1076034.1076106.

[58] Tie-Yan Liu. Learning to rank for information retrieval. *Found. Trends Inf. Retr.*, 3(3):225–331, March 2009. ISSN 1554-0669. doi: 10.1561/1500000016. URL http://dx.doi.org/10.1561/1500000016.

[59] Hang Li. *Learning to Rank for Information Retrieval and Natural Language Processing*. Morgan & Claypool Publishers, 1 edition, April 2011. ISBN 1608457079.

[60] Norbert Fuhr. Optimum polynomial retrieval functions based on the probability ranking principle. *ACM Trans. Inf. Syst.*, 7(3):183–204, July 1989. ISSN 1046-8188. doi: 10.1145/65943.65944. URL http://doi.acm.org/10.1145/65943.65944.

[61] David Cossock and Tong Zhang. Subset ranking using regression. In Gábor Lugosi and Hans Ulrich Simon, editors, *Learning Theory*, number 4005 in Lecture Notes in Computer Science, pages 605–619. Springer Berlin Heidelberg, January 2006. ISBN 978-3-540-35294-5, 978-3-540-35296-9. URL http://link.springer.com/chapter/10.1007/11776420_44.

[62] Koby Crammer and Yoram Singer. Pranking with ranking. In *NIPS*, volume 14, page 641–647, 2001.

[63] Ping Li, Qiang Wu, and Christopher J. C. Burges. McRank: learning to rank using multiple classification and gradient boosting. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems 20*, page 897–904. Curran Associates, Inc., 2008. URL http://papers.nips.cc/paper/3270-mcrank-learning-to-rank-using-multiple-classification-and-gradient-boosting.pdf.

[64] Yunbo Cao, Jun Xu, Tie-Yan Liu, Hang Li, Yalou Huang, and Hsiao-Wuen Hon. Adapting ranking SVM to document retrieval. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '06, page 186–193, New York, NY, USA, 2006. ACM. ISBN 1-59593-369-7. doi: 10.1145/1148170.1148205. URL http://doi.acm.org/10.1145/1148170.1148205.

[65] Yoav Freund, Raj Iyer, Robert E. Schapire, and Yoram Singer. An efficient boosting algorithm for combining preferences. *J. Mach. Learn. Res.*, 4:933–969, December 2003. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=945365.964285.

[66] Christopher J. C. Burges. From RankNet to LambdaRank to LambdaMART: an overview. Technical report, Microsoft Research, 2010. URL http://research.microsoft.com/en-us/um/people/cburges/tech_reports/MSR-TR-2010-82.pdf.

[67] Christopher J. C. Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In *Proceedings of the 22Nd International Conference on Machine Learning*, ICML '05, page 89–96, New York, NY, USA, 2005. ACM. ISBN 1-59593-180-5. doi: 10.1145/1102351.1102363. URL http://doi.acm.org/10.1145/1102351.1102363.

[68] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to rank: From pairwise approach to listwise approach. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, page 129–136, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-793-3. doi: 10.1145/1273496.1273513. URL http://doi.acm.org/10.1145/1273496.1273513.

[69] Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, and Hang Li. Listwise approach to learning to rank: Theory and algorithm. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, page 1192–1199, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-205-4. doi: 10.1145/1390156.1390306. URL http://doi.acm.org/10.1145/1390156.1390306.

[70] Jun Xu and Hang Li. AdaRank: a boosting algorithm for information retrieval. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research*

and Development in Information Retrieval, SIGIR '07, page 391–398, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-597-7. doi: 10.1145/1277741.1277809. URL http://doi.acm.org/10.1145/1277741.1277809.

[71] Yisong Yue, Thomas Finley, Filip Radlinski, and Thorsten Joachims. A support vector method for optimizing average precision. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '07, page 271–278, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-597-7. doi: 10.1145/1277741.1277790. URL http://doi.acm.org/10.1145/1277741.1277790.

[72] Tie-Yan Liu. *Learning to Rank for Information Retrieval*. Springer, 2011 edition, May 2011. ISBN 3642142664.

[73] Craig Macdonald, Rodrygo L. T. Santos, and Iadh Ounis. The whens and hows of learning to rank for web search. *Information Retrieval*, 16(5):584–628, October 2013. ISSN 1386-4564, 1573-7659. doi: 10.1007/s10791-012-9209-9. URL http://link.springer.com/article/10.1007/s10791-012-9209-9.

[74] Van Dang, Michael Bendersky, and W. Bruce Croft. Two-stage learning to rank for information retrieval. In Pavel Serdyukov, Pavel Braslavski, Sergei O. Kuznetsov, Jaap Kamps, Stefan Rüger, Eugene Agichtein, Ilya Segalovich, and Emine Yilmaz, editors, *Advances in Information Retrieval ECIR 2013*, number 7814 in Lecture Notes in Computer Science, pages 423–434. Springer Berlin Heidelberg, January 2013. ISBN 978-3-642-36972-8, 978-3-642-36973-5. URL http://link.springer.com/chapter/10.1007/978-3-642-36973-5_36.

[75] Craig Macdonald and Iadh Ounis. Combining fields in known-item email search. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 675–676, Seattle, Washington, USA, 2006. ACM. ISBN 1-59593-369-7. doi: 10.1145/1148170.1148312. URL http://portal.acm.org/citation.cfm?id=1148170.1148312.

[76] Jinyoung Kim, Xiaobing Xue, and W. Bruce Croft. A probabilistic retrieval model for semistructured data. In Mohand Boughanem, Catherine Berrut, Josiane Mothe, and Chantal Soule-Dupuy, editors, *Advances in Information Retrieval ECIR 2009*, number 5478 in Lecture Notes in Computer Science, pages 228–239. Springer Berlin Heidelberg, January 2009. ISBN 978-3-642-00957-0, 978-3-642-00958-7. URL http://link.springer.com/chapter/10.1007/978-3-642-00958-7_22.

[77] Stephen E. Robertson and Hugo Zaragoza. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389, 2009. ISSN 1554-0669. doi: 10.1561/1500000019. URL http://dx.doi.org/10.1561/1500000019.

[78] Craig Macdonald. *The Voting Model for People Search*. PhD dissertation, University of Glasgow, Glasgow, Scotland, UK, 2009.

[79] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, May 1983. ISSN 0036-8075, 1095-9203. doi: 10.1126/science.220.4598.671. URL http://www.sciencemag.org/content/220/4598/671.

[80] Inderjit S. Dhillon, Yuqiang Guan, and Brian Kulis. Weighted graph cuts without eigenvectors a multilevel approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(11):1944–1957, 2007. ISSN 0162-8828. doi: 10.1109/TPAMI.2007.1115.

[81] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank citation ranking: Bringing order to the web. Technical Report 1999-0120, Computer Science Department, Standford University, 1999. URL `http://ilpubs.stanford.edu:8090/422/1/1999-66.pdf`.

[82] Howard D. White. Combining bibliometrics, information retrieval, and relevance theory, part 2: Some implications for information science. *Journal of the American Society for Information Science and Technology*, 58(4):583–605, 2007. URL `http://dx.doi.org/10.1002/asi.20542`.

[83] Qiang Wu, Christopher J. C. Burges, Krysta M. Svore, and Jianfeng Gao. Adapting boosting for information retrieval measures. *Information Retrieval*, 13(3):254–270, June 2010. ISSN 1386-4564, 1573-7659. doi: 10.1007/s10791-009-9112-1. URL `http://link.springer.com/article/10.1007/s10791-009-9112-1`.

[84] Donald Metzler and W. Bruce Croft. Linear feature-based models for information retrieval. *Information Retrieval*, 10(3):257–274, June 2007. ISSN 1386-4564, 1573-7659. doi: 10.1007/s10791-006-9019-z. URL `http://link.springer.com/article/10.1007/s10791-006-9019-z`.

[85] Christopher J. C. Burges, Krysta M. Svore, Paul N. Bennett, Andrzej Pastusiak, and Qiang Wu. Learning to rank using an ensemble of lambda-gradient models. *Journal of Machine Learning Research: Workshop and Conference Proceedings*, 14:25–35, 2011.

[86] Yasser Ganjisaffar, Rich Caruana, and Cristina Videira Lopes. Bagging gradient-boosted trees for high precision, low variance ranking models. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, page 85–94, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0757-4. doi: 10.1145/2009916.2009932. URL `http://doi.acm.org/10.1145/2009916.2009932`.

[87] Craig Macdonald, Rodrygo L. T. Santos, Iadh Ounis, and Ben He. About learning models with multiple query-dependent features. *ACM Trans. Inf. Syst.*, 31(3):11:1–11:39, August 2013. ISSN 1046-8188. doi: 10.1145/2493175.2493176. URL `http://doi.acm.org/10.1145/2493175.2493176`.

[88] Steven Bethard and Daniel Jurafsky. Who should i cite: Learning literature search models from citation behavior. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, page 609–618, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0099-5. doi: 10.1145/1871437.1871517. URL `http://doi.acm.org/10.1145/1871437.1871517`.

Vita

Haozhen Zhao

College of Computing & Informatics
Drexel University
3141 Chestnut Street
Philadelphia, PA 19104 USA

Email: zhaohaozhen@gmail.com
Homepage: http://haozhenzhao.com/

- Ph.D. in Information Studies, Drexel University, USA, 2008-2015.
  - Advisors: Xiaohua (Tony) Hu (2013.6-2015.9); Xia Lin (2008.9-2013.5)
- M.A. in Information Science, Wuhan University, China, 2005-2008.
  - Thesis: *Semantic Information Extraction from Chinese Wikipedia*
- B.A. in Information Systems, Wuhan University, China, 2002-2005.

Selected Publications

- Haozhen Zhao. Sharding for literature search via cutting citation graphs. In *Big Data (Big Data), 2014 IEEE International Conference on*, pages 77–79, 2014.
- Haozhen Zhao and Xiaohua Hu. Language model document priors based on citation and co-citation analysis. In *BIR 2014: ECIR Workshop on Bibliometric-enhanced Information Retrieval*, 2014.
- Haozhen Zhao and Xiaohua Hu. Drexel at TREC 2014 federated web search track. In *TREC 2014*, 2014.
- Xia Lin, Mi Zhang, Haozhen Zhao, and Jan Buzydlowski. Multi-view of the ACM classification system. In *Proceedings of the 12th ACM/IEEE-CS Joint Conference on Digital Libraries*, JCDL '12, page 397–398, New York, NY, USA, 2012. ACM.
- Haozhen Zhao and Xia Lin. A comparison of mapping algorithms for author co-citation data analysis. *Proceedings of the American Society for Information Science and Technology*, 47(1):1–3, 2010.
- Robin A Naughton, Catherine E Hall, Haozhen Zhao, and Xia Lin. PhD portal: developing an online iSchool doctoral student community. *Proceedings of iConference 2010*, 2010.
- Daqing He, Lorri Mon, Jeffrey Pomerantz, and Haozhen Zhao. Developing a collaborative sandbox for digital library research. *2010 iConference*, page 350, 2010.
- Thomas H Park, Jiexun Li, Haozhen Zhao, and Michael Chau. Analyzing writing styles of bloggers with different opinions. *Proceedings of the 19th Annual Workshop on Information Technologies and Systems*, page 151–156, 2009.
- Xia Lin and Haozhen Zhao. Layered cocitation networks. In *5th International Conference on Webometrics, Informetrics and Scientometrics & 10th COLLNET Meeting*, Dalian, China, 2009.
- Haozhen Zhao, Tom Casteel, and Xia Lin. A dynamic visualization interface for search service. In *Proceedings of iConference 2009*, UNC, February 2009.