Web Mining Techniques for Query Log Analysis and Expertise Retrieval

DENG, Hongbo

A Thesis Submitted in Partial Fulfilment
of the Requirements for the Degree of
Doctor of Philosophy
in
Computer Science and Engineering

The Chinese University of Hong Kong
September 2009

Thesis/Assessment Committee Members

Professor Kwong-Sak LEUNG (Chair)

Professor Irwin KING (Thesis Supervisor)

Professor Michael R. LYU (Thesis Supervisor)

Professor Lai-Wan CHAN (Committee Member)

Professor Qiang YANG (External Examiner)

Abstract of thesis entitled:

Web Mining Techniques for Query Log Analysis and Expertise Retrieval
Submitted by DENG, Hongbo
for the degree of Doctor of Philosophy
at The Chinese University of Hong Kong in September 2009

With the large increase in the amount of information available online, rich Web data can be obtained on the Internet, such as over one trillion Web pages, millions of scientific literature, and different interactions with society, like question answers, query logs. Currently, Web mining techniques has emerged as an important research area to help Web users find their information need. In general, Web users express their information need as queries, and expect to obtain the needed information from the Web data through Web mining techniques. To better understand what users want in terms of the given query, it is very essential to analyze the query logs. On the other hand, the returned information may be Web pages, images, and other types of data. Beyond the traditional information, it would be quite interesting and important to identify relevant experts with expertise for further consulting about the query topic, which is also called expertise retrieval.

The objective of this thesis is to establish automatic content analysis methods and scalable graph-based models for query log analysis and expertise retrieval. One important aspect of this thesis is therefore to develop a framework to combine the content information and the graph information with the following two purposes: 1) analyzing Web contents with graph structures,

more specifically, mining query logs; and 2) identifying high-level information needs, such as expertise retrieval, behind the contents.

For the first purpose, a novel entropy-biased framework is proposed for modeling bipartite graphs, which is applied to the click graph for better query representation by treating heterogeneous query-URL pairs differently and diminishing the effect of noisy links. Based on the graph information, there is a lack of constraints to make sure the final relevance of the score propagation on the graph. To tackle this problem, a general Co-HITS algorithm is developed to incorporate the bipartite graph with the content information from both sides as well as the constraints of relevance. Extensive evaluations on query log analysis demonstrate the effectiveness of the proposed models.

For the second purpose, a weighted language model is proposed to aggregate the expertise of a candidate from the associated documents. The model not only considers the relevance of documents against a given query, but also incorporates important factors of the documents in the form of document priors. Moreover, an important approach is presented to boost the expertise retrieve by incorporating the content with other implicit link information through the graph-based re-ranking model. Furthermore, two community-aware strategies are developed and investigated to enhance the expertise retrieval, which are motivated by the observation that communities could provide valuable insight and distinctive information. Experimental results on the expert finding task demonstrate these methods can improve and enhance traditional the traditional expertise retrieval models with better performance.

論文題目： Web 挖掘技術及其在搜索引擎查詢日誌和專家搜索中的應用

作者 ： 鄧洪波

學校 ： 香港中文大學

學系 ： 計算器科學及工程學系

修讀學位： 哲學博士

摘要 ：

隨著網上信息量的大量增加，網絡用戶正被過多的資訊所淹沒並且面臨資訊過載的現象。為了幫助用戶找到需要的信息，一個關鍵問題是根據所提供的查詢了解用戶想要什麼，而另一個重要的問題查找相關的具有專業知識的專家以便作進一步咨詢。爲實現以上目標，Web 挖掘已發展為一個重要的跨學科研究領域，並且通過利用信息檢索、數據挖掘和機器學習進行查詢日誌分析和專家檢索。

本論文通过建立自動內容分析的方法和可擴展的基於圖的模型，以识别所需要的具有高相關性和高質量的信息資訊。該輪文的一個重要方面是結合資訊內容和圖結構兩方面的信息以制定一個基本框架來達到以下兩個目的：1）分析具有圖結構的搜索引擎查詢日誌；和 2）查找高級別的信息需求，如具有專業知識的專家檢索。

為了第一個目的，我們提出一種新的熵偏置的框架對二部圖進行建模，並且應用到具有二部圖結構的查詢日誌單擊圖。這一模型可以減少二部圖中噪聲鏈接的影響和區別加權單擊圖中具有不同特性的鏈接。基於圖的模型可以捕捉不同節點之間的語義相似性，但是如果只利用圖的信息，將會缺乏內容相關性的制約。為解

決這一問題，我們提出一種通用的 Co-HITS 算法，這個算法充分利用二部圖及其兩邊的內容信息來提供內容相關性的限制。實驗結果表明這兩種方法能有效地應用于查詢日誌分析並改善其性能。

為了第二個目的，我們提出一種加權統計語言模型通過分析相關文件來匯總候選人的專門知識。對於給定的查詢，目標是找到該查詢專業的專家。這個模型不但考慮了文件的相關性，還考慮了文件的重要性。此外，本文通過基於圖的重排序模型綜合考慮了文件的內容和隱含的鏈接信息，從而達到加強專家搜索性能的目的。由於相關專家之間可能形成相應的社區，這個信息將對於提升專家搜索的性能非常有幫助，因此我們提出了兩個基於社區信息的策略來增強專家搜索的可靠性。實驗結果表明這幾種方法能有效地改善傳統的專家搜索模型的性能。

Acknowledgement

I would like to express my greatest thanks to my supervisors, Prof. Irwin King and Prof. Michael R. Lyu. Their patient guidance, encouragement and advice have been absolutely essential for the completion of this thesis. Over the years, I have learned many many things from them, not only on research, but also about presentation, technical skills and writing. Moreover, I learned to truly appreciate high quality in research.

I am grateful to my thesis committee members, Prof. L.W. Chan and Prof. K.S. Leung for their helpful comments and suggestions about this thesis. My special thanks to Prof. Qiang Yang who kindly served as the external committee for this thesis. I would like to thank my mentor, Dr. Lei Zhang, for his guidance and support when I was an intern in Microsoft Research Asia. I extend my thanks to Dr. Hugo Zaragoza for the conservations and discussions when I was visiting Yahoo! Research Barcelona.

I would also like to thank my colleagues in the machine learning and web intelligence group, Haixuan Yang, Kaizhu Huang, Zenglin Xu, Allen Lin, Hao Ma, Haiqin Yang, Xin Xin and Chao Zhou. I also thank my officemates and friends, Xinyu Chen, Xiaoqi Li, Jianke Zhu, Yangfan Zhou, Wujie Zheng, Zibin Zheng, Junjie Xiong, Dan Wu, Yingyi Bu, David Liu and many others, who made me have a relaxed and happy time at CUHK.

Finally, I am deeply grateful to my parents and family for their unlimited love and strong support for my graduate study.

To my beloved parents and Rong.

Contents

Abstract **i**

Acknowledgement **v**

1 Introduction **1**

1.1 Overview 1

1.2 Web Mining Techniques 3

1.2.1 Traditional Information Retrieval 3

1.2.2 Link Analysis 4

1.2.3 Machine Learning 5

1.3 Applications 6

1.3.1 Query Log Analysis 6

1.3.2 Expertise Retrieval 7

1.4 The Unified Framework and Its Contributions 8

1.5 Thesis Organization 13

2 Background Review **16**

2.1 Information Retrieval Models 16

2.1.1 Vector Space Model 17

2.1.2 Probabilistic Model 18

2.2 Web Link Analysis 21

2.2.1 PageRank 21

2.2.2 HITS 22
2.2.3 Other Variations 23
2.3 Semi-supervised Learning 24
2.4 Query Log Analysis 25
2.5 Expertise Retrieval 27

3 Entropy-biased Models for Click Graphs 31
3.1 Problem and Motivation 32
3.2 Query Representation Models 35
3.2.1 Preliminaries and Notations 35
3.2.2 Click Frequency Model 38
3.2.3 Entropy-biased Model 39
3.2.4 Connection with Other Methods 43
3.3 Mining Query Log on Click Graph 44
3.3.1 Query-to-Query Similarity Measurement 44
3.3.2 Graph-based Random Walk Model 45
3.4 Experimental Evaluation 46
3.4.1 Data Collection and Analysis 46
3.4.2 Assessments and Evaluation Metrics 47
3.4.3 Query Similarity Analysis 50
3.4.4 Random Walk Evaluation 54
3.5 Summary 56

4 Generalized Co-HITS Algorithm 58
4.1 Problem and Motivation 59
4.2 Preliminaries 62
4.3 Generalized Co-HITS Algorithm 63
4.3.1 Iterative Framework 63
4.3.2 Regularization Framework 65

4.3.3 Connections and Justifications 69
4.4 Application to Bipartite Graphs 70
4.4.1 Bipartite Graph Construction 70
4.4.2 Statistical Language Model 71
4.4.3 Overall Algorithm 72
4.5 Experimental Evaluation 73
4.5.1 Data Collection 74
4.5.2 Assessments and Evaluation Metrics 75
4.5.3 Experimental Results 76
4.6 Related Work and Discussions 82
4.7 Summary 84

5 Modeling Expertise Retrieval **86**
5.1 Problem and Motivation 87
5.2 Modeling Expert Search 91
5.2.1 Problem Definition 92
5.2.2 Paper Relevance 93
5.2.3 Paper Importance 95
5.2.4 Expertise Aggregation 96
5.3 Graph-based Regularization 98
5.3.1 Regularization Framework 99
5.3.2 Graph Construction 100
5.3.3 Connections and Justifications 101
5.4 Experimental Setup 103
5.4.1 DBLP Collection and Representation 103
5.4.2 Assessments 107
5.4.3 Evaluation Metrics 108
5.5 Experimental Results 110
5.5.1 Preliminary Experiments 110

5.5.2 Language Models with Paper Importance 111
5.5.3 Effect of Graph-based Regularization 114
5.5.4 Comparison and Detailed Results 115
5.6 Summary 116

6 Enhancing Expertise Retrieval **121**
6.1 Motivation 121
6.2 Preliminaries 125
6.3 Document-based Models with Community-Aware Smoothing . 126
6.3.1 Statistical Language Model 127
6.3.2 Smoothing Using Community Context 128
6.3.3 Determining Other Probabilities 129
6.4 Enhanced Models with Community-Aware Authorities 131
6.4.1 Discovering Authorities in a Community 132
6.4.2 Community-Sensitive AuthorRank 133
6.4.3 Ranking Refinement Strategy 135
6.4.4 Overall Algorithm 136
6.5 Experimental Evaluation 137
6.5.1 Dataset 137
6.5.2 Assessments and Evaluation Metrics 138
6.5.3 Experimental Results 141
6.6 Summary 149

7 Conclusions **151**
7.1 Summary 151
7.2 Future Work 153

A List of Publications **156**

Bibliography **158**

List of Figures

1.1 The query log data. 9
1.2 The expertise retrieval data. 10

3.1 Example of a click graph. 32
3.2 The click frequency from the query "map". 33
3.3 The surface specified by the click frequency, query frequency and cfiqf, with color specified by the cfiqf value. The color is proportional to the surface height. 42
3.4 The distributions of the (a) click frequency, (b) user frequency and (c) query frequency. 48
3.5 The performance comparison of six models (CF, CF-IQF, UF, UF-IQF, TF and TF-IDF models) using two different similarity measurements. 51
3.6 The performance of random walk model. 56

4.1 Example of a bipartite graph. The edges between U and V are represented as the transition matrices W^{uv} and W^{vu}. Note that the dashed lines represent hidden links when considering the vertices in one side, where W^{uu} and W^{vv} denote the hidden transition matrices within U and V respectively. 60
4.2 Score propagation on the bipartite graph: (a) score y_k is propagated to u_i and u_j, and (b) score x_i is propagated to v_k. . . . 64

4.3 The effect of varying parameters (λ_u and λ_v) in the iteration framework: (a) personalized PageRank, (b) one-step propagation, and (c) general Co-HITS. The dashed lines denote the baseline results. 77
4.4 The effect of varying parameters (μ_α and λ_r) in the regularization framework: (a) single-sided regularization, and (b) double-sided regularization. 79
4.5 Comparison of six models. 82

5.1 A sample of the artificial expert search process. 88
5.2 The schematic of general expert finding systems. 91
5.3 The weighted model for expert finding. 93
5.4 A query example with documents and authors. 95
5.5 A sample of the DBLP XML records. 103
5.6 A snapshot for the search results of Google Scholar. 104
5.7 The representation of a document. After crawling and parsing the search results from Google Scholar, we combine the paper title d_T and the supplemental data d_S as the representation of a document. 105
5.8 The effect of varying the parameter μ_α by comparing four different models, including the basic language model *LM(bas)*, basic language model with graph-based regularization *LM(r)*, weighted language model *LM(w)*, and its extension with graph-based regularization *LM(w+r)*. 113
5.9 Illustration and comparison of the experimental results on each query for four different methods. 119

6.1 An example graph with two communities 123
6.2 The document-based model for expertise retrieval. 126

6.3 A graph representation of the relationships between documents, communities and the entire collection. 129
6.4 Coauthorship graph with: (a) coauthorship frequency, and (b) normalized weight. 133
6.5 An example of the DBLP XML records. 138
6.6 The effect of varying the parameters (k_1 and k_2) in (a) the document-based model $DM(wc)$ and (b) the enhanced model $EDM(wc)$. 146

List of Tables

1.1 Several developed models within the framework. 10

3.1 Click frequency matrix for the example click graph. 36
3.2 Table of Notation. 37
3.3 CF transition probabilities for the example click graph. 38
3.4 IQF values of the URLs 41
3.5 CF-IQF transition probabilities for the example click graph. . 43
3.6 Samples of the AOL query log dataset. 47
3.7 Comparison of different methods by P@1 and P@10. We also show the percentage of relative improvement in the lower part. 52
3.8 Examples of query suggestions generated by two different models on click graph. 55

4.1 Connections with other methods 68
4.2 Samples of the AOL query log dataset. 74
4.3 Comparison of different methods by P@5 and P@10. The mean precisions and the percentages of relative improvements are shown in the table. 81

5.1 Combination of different methods. 98
5.2 Statistics of the DBLP collection. 107
5.3 Benchmark dataset of 17 queries. 109
5.4 Experimental results with different representations (%). 111

5.5 Evaluation results of language models using different weighting methods (%). Best scores are in boldface. 112
5.6 Comparison of different methods (%). The percentages of relative improvements are shown in the lower part. 117

6.1 Combination of different methods. 130
6.2 Statistics of the DBLP collection. 139
6.3 Benchmark dataset of 17 queries. 140
6.4 Comparison of different document-based methods. The percentages of relative improvements are shown in the lower part. 143
6.5 Comparison of different enhanced methods. The percentages of relative improvements are shown in the lower part. 145
6.6 The detailed results of the community-sensitive AuthorRank for the query "machine learning." The first row is the top-5 communities for the query, and the rest part lists the top-10 author lists ranked by their authorities in the community. . . . 147
6.7 The top-10 expert lists retrieved by the document-based model $DM(wc)$, the community-sensitive AuthorRank, and the enhanced model $EDM(wc)$, for the query "machine learning." . 148

Chapter 1

Introduction

1.1 Overview

The World Wide Web (Web) has been providing an important and indispensable platform for receiving information and disseminating information as well as interacting with society on the Internet. With its astronomical growth over the past decade, the Web becomes huge, diverse, and dynamic. On July 25, 2008, Google software engineers Jesse Alpert and Nissan Hajaj announced that Google Search[1] had discovered one trillion unique URLs [4]. Due to the properties of the Web data, we are currently drowning in information and facing information overload [90]. The information may consist of Web pages, images, people and other types of data. *To help Web users find their information need, a critical issue is to understand what users want with respect to the given query by mining the query logs. On the other hand, it would be quite interesting and important to identify relevant experts with expertise for further consulting about the query topic, which is also called expertise retrieval.* In order to achieve the above goals, Web mining has emerged as an important interdisciplinary research area by leveraging several disciplines such as

[1] `http://www.google.com/`

information retrieval, data mining, machine learning, and database systems.

The Web mining research field is fast moving, and has encountered a great number of challenges such as scalability, spam, content quality, unstructured data and so on [55]. As a result, many research efforts have been devoted to pushing forward the techniques for Web search and mining. According to analysis targets, these methods can be divided into three different types, including Web content mining, Web structure mining and Web usage mining [71, 82]. Basically, the Web content consists of several types of data such as textual, image, audio, etc. Web content mining sometimes is called text mining, because much of the Web content data is unstructured text data [23, 47], which could be used to measure the relevance to the needed information based on information retrieval models. As for the Web structure mining, it is the process of using link analysis algorithms to analyze the node and discover the model from link structures of the Web [24]. Web usage mining [61, 99, 105, 135] try to make sense of the data generated by the Web surfer's sessions or behaviors. Since the content and the link structure are two essential and important parts for the Web data, there is an increasing demand to develop more advanced models by mining multiple information sources, especially the text (content) and link structure (graph) information, so as to identify needed information with high relevance and quality.

The objective of this thesis is to establish automatic content analysis methods and scalable graph-based models for identifying the needed information with high relevance and good quality. Many data types arising from Web search and mining applications can be modeled as the combination of both content and graph information. Examples include queries and URLs in query logs, and authors and papers in scientific literature. One important aspect of this thesis is therefore to develop a framework to combine the content information and the graph information with the following two purposes: 1) analyzing

Web contents with graph structures, more specifically, mining query logs; and 2) identifying high-level information needs, such as expertise retrieval, behind the contents. As the query log is a good resource that records users' search histories, it is very essential to mine the query logs for capturing users' information needs. In addition, expertise retrieval can be viewed as a high-level information retrieval beyond the traditional document retrieval, whose task is to retrieve a ranked list of persons who possess expertise on a given topic. In this thesis, several general models are proposed for query log analysis and expertise retrieval.

In this chapter, we briefly introduce the Web mining techniques as well as the applications, including the query log analysis and expertise retrieval. Then we present the objectives of this thesis and outline the contributions. Finally, we provide an overview of the rest of this thesis.

1.2 Web Mining Techniques

The Web search and mining research is a converging research area from several communities, such as *information retrieval*, *link analysis*, *data mining*, and *machine learning*, as well as others. Each of them has been separately studied in the past decades. Let us briefly introduce them as follows.

1.2.1 Traditional Information Retrieval

Web search and mining has its root in information retrieval [5, 82, 91]. In general, information retrieval (IR) refers to the retrieval of unstructured data. Most often, it is related to Text Retrieval, i.e. the retrieval of textual documents. Other types of retrieval include, for example, Image Retrieval, Video Retrieval, and Music Retrieval. Retrieving information simply means finding a set of documents that are relevant to the user query. Clearly, one central problem regarding information retrieval systems is to rank documents

optimally given a query so that relevant documents would be ranked above nonrelevant ones. The retrieval accuracy of an IR system is directly determined by the quality of the scoring function. Thus, a major research challenge in information retrieval is to seek an optimal scoring function (retrieval function), which is based on a retrieval model. Many important IR models based on the content information have been proposed to derive the retrieval functions that can be computed to score and rank documents. We will briefly introduce some traditional IR models in Chapter 2.1.

1.2.2 Link Analysis

The analysis of hyperlinks and the graph structure of the Web has been instrumental in the development of Web search [91]. Link analysis [17, 22] is one of many factors considered by Web search engines in computing a composite score for a Web page on any given query. Basically, link analysis for Web search has intellectual antecedents in the field of citation analysis [50, 119], which seeks to quantify the influence of scholarly articles by analyzing the pattern of citations among them. As citations represent the conferral of authority from a scholarly article to others, link analysis on the Web treats hyperlinks from a Web page to another as a conferral of authority. The phenomenon of citation is prevalent and dependable enough that it is feasible for Web search engines to derive useful signals for ranking from more sophisticated link analysis. Several Web search ranking algorithms use link-based centrality metrics, including Marchiori's Hyper Search [92], Google's PageRank [18], Kleinberg's HITS algorithm [69], and the TrustRank algorithm [52]. Furthermore, link analysis is also conducted in social network analysis in order to understand and extract information from the relationships between individual in social networks. Such individuals are often persons, but may be groups, organizations, nation states, Web sites, or citations between scholarly

publications. For example the analysis might be of the interlinking between researchers, politicians' Web sites or blogs. We will briefly introduce some link analysis methods in Chapter 2.2.

1.2.3 Machine Learning

There is a close relationship between machine learning and Web mining research areas. A major focus of machine learning research is to learn to recognize complex patterns and make decisions based on data. Machine learning has been applied to many applications of the Web search and mining, such as learning to rank [3, 20, 109], text categorization [62, 77, 124, 143], Web query classification [12, 81, 129], etc. In short, Web search and mining intersects with the application of machine learning on the Web.

In general, machine learning can be typically categorized as supervised learning, unsupervised learning, semi-supervised learning, as well as others. Supervised learning considers the problems of estimating certain functions from examples with label information, such as Support Vector Machines (SVM) [29, 80], Neural Network [25, 70, 76] and naive Bayes classifier [94]. Unsupervised learning considers the problem of learning from a collection of data instances without training labels. One of the most popular areas of study in unsupervised learning is data clustering techniques, which have been widely used for data mining applications [58]. Semi-supervised learning has recently been proposed to take advantage of both labeled and unlabeled data, which has been demonstrated to be a promising approach. We will introduce some semi-supervised learning methods, especially the graph-based semi-supervised learning algorithms, in Chapter 2.3.

1.3 Applications

In addition to the studies of Web mining techniques, this thesis also investigates these techniques and algorithms with applications to real-world problems. Two main applications are studied. One is query log analysis, and the other is expertise retrieval. Although there are many differences between query log analysis and expertise retrieval, the key point is that all of these data, the query log data and the expertise retrieval data, can be viewed as the combination of the content and graph information. The objective of this work is to propose a general Web mining framework to combine the content and the graph information effectively, by leveraging Web mining techniques to boost the performance of these applications. Let us briefly introduce the main applications and related problems which will be explored in this thesis.

1.3.1 Query Log Analysis

Web query log analysis has been studied widely with different Web mining techniques for improving search engines' efficacy and usability in recent years. Such studies mined the logs to improve numerous search engine's capabilities, such as query suggestion, query classification, ranking, targeted advertising, etc. The *click graph* [31], a bipartite graph between queries and URLs, is an important technique for describing the information contained in the query logs, in which edges connect a query with the URLs that were clicked by users as a result. As the edges of the click graph can capture some semantic relations between queries and URLs, it is useful to represent the query using the vector of documents when only considering the graph information. State-of-the-art approaches based on the raw click frequencies for modeling the click graph, however, are not noise-eliminated. Nor do they handle heterogeneous query-URL pairs well. To deal with these critical problems, a novel entropy-biased framework [39] is proposed for query representation on the

click graph, which incorporates raw click frequencies and other information with the entropy information of the connected URLs.

Based on the click graph, there is a natural random walk on the bipartite graph, which demonstrates certain advantages comparing with the traditional approaches based on the content information. Many link analysis methods have been proposed, such as HITS [69] and PageRank [18], to capture some semantic relations within the bipartite graph. However, there is a lack of constraints to make sure the final relevance of the score propagation on the graph, as there are many noisy edges within the bipartite graph. In this thesis, a novel and general Co-HITS algorithm [41] is proposed to incorporate the bipartite graph with the content information from both sides as well as the constraints of relevance. Moreover, the Co-HITS algorithm is investigated from two different perspectives, and applied to query suggestion by mining the query log data.

1.3.2 Expertise Retrieval

In previous subsection, we have briefly described several models and their applications to query log analysis by combining the content and graph information. As we know, there are many other data types can be regarded as the combination of the content and graph information. In this subsection, we will introduce another application, expertise retrieval, by extending the previous models to incorporate different information in a more heterogeneous information environment.

With the development of Web mining and information retrieval techniques, many research efforts in this field have been made to address high-level information retrieval and not just the traditional document retrieval, such as expertise retrieval [9]. Expertise retrieval has received increased interests since the introduction of an expert finding task in TREC 2005 [30, 133]. The

task of expertise retrieval is to identify a set of persons with relevant expertise for the given query. Traditionally, the expertise of a person is characterized based on the documents that have been associated with the person. One of the state-of-the-art approaches [8, 37] is the document-based model using a statistical language model to rank experts. However, these methods only consider the documents associated with the experts. Actually, in addition to the associated documents, there is much other information that can be included, such as the importance of the documents, the graph information, and the community information. Therefore, how to utilize these information to model and enhance the expertise retrieval becomes an interesting and challenging problem.

In this thesis, a weighted language model [37] is proposed to aggregate the expertise of a candidate from the associated documents. The model not only considers the relevance of documents against a given query, but also incorporates important factors of the documents in the form of document priors. Moreover, an important approach is presented to boost the expertise retrieval by incorporating the content with other implicit link information through the graph-based re-ranking model [40]. Furthermore, two community-aware strategies are developed and investigated to enhance the expertise retrieval, which are motivated by the observation that communities could provide valuable insight and distinctive information. Experimental results on the expert finding task demonstrate these methods can improve and enhance traditional the traditional expertise retrieval models with better performance.

1.4 The Unified Framework and Its Contributions

This thesis aims to propose a general Web mining framework to combine the content and the graph information effectively. Within this thesis, the framework is investigated and developed based on two different applications:

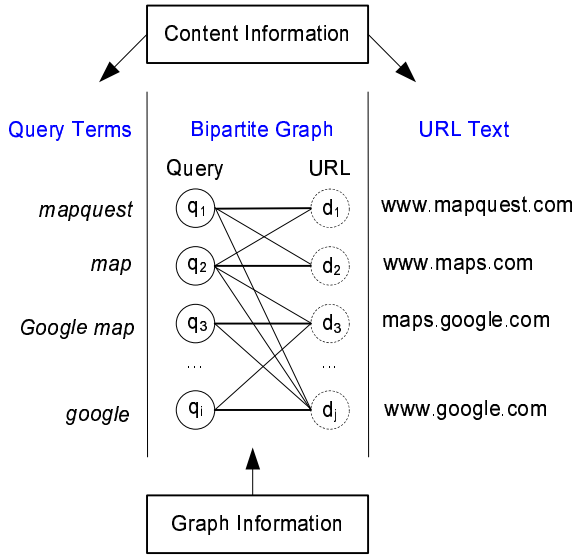


Figure 1.1: The query log data.

query log analysis and expertise retrieval. In the query log analysis, the query log data, as shown in Figure 1.1, can be modeled as a bipartite graph along with the content information, i.e., the query terms and the URL text. In the expertise retrieval, the data is more complicated as shown in Figure 1.2. Besides the paper content information, there is some individual and combined graph information, including the tripartite graph, the co-authorship graph, the citation graph, and the paper-author-community relational graph. Although there are many differences between these two applications, the key point is that both data can be viewed as the combination of the content and the graph information. Motivated by this observation, this thesis proposes a general Web mining framework to take into account the content and the graph information, which combines information retrieval models, link analysis algorithms, and machine learning techniques in a unified way to boost the performance of these applications.

Based on the framework, some challenging problems are addressed and novel models are proposed to solve them effectively. In summary, several

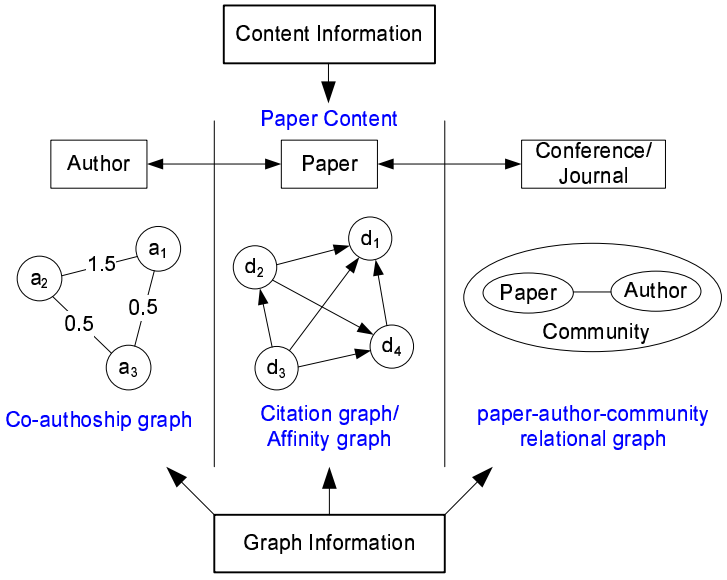


Figure 1.2: The expertise retrieval data.

Table 1.1: Several developed models within the framework.

Model	**Data**	**Techniques**
Entropy-biased Model	Bipartite Graph	LA + IR
Co-HITS Algorithm	Bipartite Graph + Content	LA + IR + ML
Weighted Language Model	Content + Citation	IR
Graph-based Re-ranking Model	Content + Affinity Graph	LA + IR +ML
Enhanced Model with Communities	Content + Community	LA + IR +ML

LA: Link Analysis

IR: Information Retrieval

ML: Machine Learning

developed models within the framework are described in Table 1.1, which illustrates the relationships between the models, the used data, and the utilized techniques. Basically, the entropy-biased model analyzes the bipartite graph using link analysis and information retrieval techniques, while the Co-HITS algorithm makes full use of the bipartite graph and the content information. A weighted language model takes into consideration not only the relevance between a query and documents but also the importance of the documents. Moreover, a graph-based re-ranking approach is developed to refine the relevance scores by regularizing the smoothness of the relevance scores on the graph along with a regularizer on the initial relevance scores. Furthermore, an enhanced model is investigated with the community information. The main contributions of this thesis can be further described as follows:

(1) **A Novel Entropy-biased Framework for Query Representation on the Click Graph.**

Based on the click graph, the query can be represented by a vector of connected documents when only considering the graph information. A novel *entropy-biased framework* is proposed for better query representation by combining the inverse query frequency with the click frequency and user frequency information simultaneously. In the framework, a new notion, namely the *inverse query frequency*, is introduced to weigh the importance of a click on a certain URL, which can be extended and used for other bipartite graphs. The proposed entropy-biased model is the *first formal model* to distinguish the variation on different query-URL pairs on the click graph. In addition, a new source, called the *user frequency*, is identified for diminishing the manipulation of the malicious clicks. In our entropy-biased framework, Click Frequency-Inverse Query Frequency (CF-IQF) is a simplified version of the entropy-biased model. And this weighting scheme can be applied to other bipartite graphs.

(2) **A Generalized Co-HITS Algorithm.**

A generalized *Co-HITS* algorithm is introduced to incorporate the bipartite graph with the content information from both sides. Moreover, the *Co-HITS* algorithm is investigated based on two frameworks, including the iterative and the regularization frameworks, which illustrate the generalized Co-HITS algorithm from different perspectives. The basic idea of the iterative framework is to propagate the scores on the bipartite graph. Compared with previous methods, the key difference is that the score is updated according to the aggregated score along with the initial relevance scores. For the iterative framework, it contains HITS and personalized PageRank as special cases. In the regularization framework, we successfully build a connection with HITS, and develop a new cost function to consider the direct relationship between two entity sets, which leads to a significant improvement over the baseline method. To illustrate the methodology, we apply the Co-HITS algorithm to the application of query suggestion by mining the query log data.

(3) **A Weighted Language Model for Expertise Retrieval with Graph-based Regularization.**

In order to investigate the combination of more heterogeneous information, the high-level expertise retrieval task is addressed based on the large-scale DBLP bibliography and its supplemental data from Google Scholar. A novel expert finding framework is proposed to identify the relevant experts in the academic field. A weighted language model is formally defined to aggregate the expertise of a candidate from the associated documents. The model takes into account not only the relevance between a query and documents, but also the importance of the documents. In the framework, the paper importance is interpreted by introducing a prior probability that the paper is written by an expert.

Furthermore, a graph-based regularization method is integrated to boost the performance by refining the relevance scores of the documents.

(4) **Enhancing Expertise Retrieval Using Community-aware Strategies.**

Motivated by the observation that communities could provide valuable insight and distinctive information, two community-aware strategies are investigated and developed to enhance the expertise retrieval. We first propose a new smoothing method using the community context for statistical language model, which is employed to identify the most relevant documents so as to reflect the expertise retrieval in the document-based model. Then, the community-sensitive AuthorRank is introduced to model the authors' authorities based on the community coauthorship networks. Finally, an adaptive ranking refinement strategy is developed to aggregate the ranking results of both document-based model and community-sensitive AuthorRank. Experimental results demonstrate the effectiveness and robustness of both community-aware strategies. Moreover, the improvements made in the enhanced models are significant and consistent.

1.5 Thesis Organization

This thesis reviews the main methodology in Web search and mining, and proposes several models that integrate different techniques to incorporate heterogeneous information simultaneously. In this thesis, several important issues, including the generalized Co-HITS algorithm and the graph-based regularization, are extensively explored to incorporate content with graph information. This thesis also extends these techniques to address some real-world problems in query log analysis and expertise retrieval applications which

demonstrate promising results. The rest of this thesis is organized as follows:

- Chapter 2

  This chapter briefly reviews some background knowledge and work related to the main methodology that will be explored in this thesis.

- Chapter 3

  This chapter studies the problem of query representation by modeling click graphs. We present a novel entropy-biased framework for modeling query representation, whose basic idea is to treat various query-URL pairs differently according to the inverse query frequency (IQF). We not only formally define and quantify this IQF weighting scheme, but also incorporate it with the click frequency and user frequency information on the click graph for an effective query representation. Extensive evaluations on query similarity analysis and query suggestion will be discussed.

- Chapter 4

  This chapter proposes a general Co-HITS algorithm to incorporate the bipartite graph with the content information from both sides. We investigate the algorithm based on two frameworks, including the iterative and the regularization frameworks. The Co-HITS algorithm is applied to the application of query suggestion, which demonstrates the effectiveness of the Co-HITS algorithm with empirical evaluation on real-world query logs.

- Chapter 5

  This chapter addresses the high-level expertise retrieval task and investigates the combination of more heterogeneous information. A novel expertise retrieval framework is proposed based on the large-scale DBLP bibliography and its supplemental data from Google Scholar. We define

a weighted language model to aggregate the expertise of a candidate from the associated documents. Moreover, we integrate a graph-based regularization method to enhance our model by refining the relevance scores of the documents with respect to the query. Empirical results on benchmark datasets will be discussed.

- Chapter 6

  This chapter describes two community-aware strategies to enhance expertise retrieval. One strategy is to smooth the language model with the community context, and the other strategy is to develop an adaptive ranking refinement method with the community-sensitive authorities. This work is motivated by the observation that communities could provide valuable insight and distinctive information. Extensive evaluation on benchmark datasets will be studied.

- Chapter 7

  The last chapter summarizes this thesis and addresses some directions to be explored in future work.

In order to make each of these chapters self-contained, some critical contents, e.g., model definitions or motivations having appeared in previous chapters, may be briefly reiterated in some chapters.

□ **End of chapter.**

Chapter 2

Background Review

In this chapter, we briefly review some backgrounds about Web mining techniques, including information retrieval models, link analysis algorithms, semi-supervised learning. In addition, we introduce the main applications, i.e., query log analysis and expertise retrieval, that will be explored in the thesis.

2.1 Information Retrieval Models

In information retrieval, a major research challenge is to seek an optimal ranking function, which is based on a retrieval model. The retrieval model formally defines the notion of relevance and enables us to derive a retrieval function that can be computed to score and rank documents. The three classic models in information retrieval are called Boolean model, vector space model, and probabilistic model. The Boolean model [51, 122] is a simple retrieval model based on set theory and Boolean algebra, in which documents and queries are represented as sets of index terms. As the Boolean model suffers from major drawbacks [5], here we mainly introduce the vector space model [65, 123, 141, 121], and the probabilistic model [74, 115, 134] especially the language model [75, 108, 145, 147, 148].

2.1.1 Vector Space Model

In the vector space model [120, 123, 141], documents and queries are represented as vectors in a t-dimensional space. Each dimension corresponds to a separate term. The definition of *term* depends on the application, which could be single word, keyword, or longer phrase. Typically, the words are chosen to be the terms, the dimensionality of the vector is the number of words in the vocabulary. If a term t occurs in the document d_j, the term is associated with a non-zero weight w_{t,d_j} in the document vector $\overrightarrow{d_j}$. These term weights are ultimately used to compute the *similarity* or *relevance* between each document and the user query.

There are several different ways developed for computing these term weights, and one of the best known schemes is TF-IDF (term frequency-inverse document frequency) weighting [65, 121]. Let $freq_{i,j}$ (raw term frequency) be the number of times a given term t_i occurs in the document d_j. Then, the normalized *term frequency* $tf_{i,j}$ of term t_i in document d_j is given by

$$tf_{i,j} = \frac{freq_{i,j}}{\sum_i freq_{i,j}}, \tag{2.1}$$

where the denominator is the sum number of all terms in document d_j. If the term t_i does not appear in the document d_i, then $tf_{i,j} = 0$. Such term frequency is provides one measure of how well that the term describes the document contents. The *inverse document frequency* is a measure of the general importance of the term. Let N be the total number of documents in the corpus and n_i be the number of documents in which the term t_i appears. Thus, the inverse document frequency for t_i is given by

$$idf_i = \log \frac{N}{n_i}. \tag{2.2}$$

The best known TF-IDF weights are given by

$$w_{i,j} = tf_{i,j} \times \log \frac{N}{n_i}. \tag{2.3}$$

A high weight in TF-IDF is reached by a high term frequency (in the given document) and a low document frequency of the term in the whole collection of documents; the weights hence tend to filter out common terms.

For a collection of documents, a term-document matrix can be utilized to describe the occurrences of terms in documents. Generally, it is a sparse matrix. In order to reduce the high dimensionality of term-document matrix, some advanced techniques, such as Latent Semantic Analysis (LSA) [36, 57], are proposed to transforms the occurrence matrix into a relation between the terms and some concepts.

For the vector space model, the document vector $\overrightarrow{d_j}$ is defined as $\overrightarrow{d_j} = (w_{1,j}, w_{2,j}, ..., w_{t,j})$, and the query vector $\overrightarrow{q_i} = (w_{1,i}, w_{2,i}, ..., w_{t,i})$. The vector space model proposes to evaluate the similarity of the document d_j with regard to the query q_i as the correlation between these two vectors $\overrightarrow{d_j}$ and $\overrightarrow{q_i}$. One simple way is to quantify the correlation using the cosine of the angle between two vectors, which is defined as

$$sim(d_j, q_i) = \frac{\overrightarrow{d_j} \cdot \overrightarrow{q_i}}{|\overrightarrow{d_j}| \times |\overrightarrow{q_i}|} = \frac{\sum_{k=1}^{t} w_{j,k} \times w_{k,i}}{\sqrt{\sum_{k=1}^{t} w_{j,k}^2} \times \sqrt{\sum_{k=1}^{t} w_{k,i}^2}} \tag{2.4}$$

where $|\overrightarrow{d_j}|$ and $|\overrightarrow{q_i}|$ are the norms of the document and query vectors. The factor $|\overrightarrow{q_i}|$ does not affect the ranking of documents because it is the same for all documents, while the factor $|\overrightarrow{d_j}|$ provides a normalization in the space of the documents. In general, the vector space model with heuristic TF-IDF weighting and document length normalization [131] has traditionally been one of the most effective retrieval models, and it remains quite competitive as a popular retrieval model.

2.1.2 Probabilistic Model

In the probability model, the process of document retrieval can be treated as estimating the probability that this document is relevant to this query [74,

115, 134]. The Probability Ranking Principle proposed in [112] is often taken as the foundation for probabilistic retrieval models. Formally, let R be a binary random variable that includes whether d is relevant to q or not. It takes two values which we denotes as r ("relevant") and $\overline{r}$ ("not relevant"). Given a query q, the probabilistic model assigns to each document d_j the ratio $p(r|q,d_j)/p(\overline{r}|q,d_j)$ which computes the odds of the document d_j being relevant to the query q. Equivalently, we may use the following log-odds ratio to rank documents:

$$\log \frac{p(r|q,d_j)}{p(\overline{r}|q,d_j)} = \log \frac{p(q,d_j|r)p(r)}{p(q,d_j|\overline{r})p(\overline{r})}. \tag{2.5}$$

Two different ways are reviewed in [145] to factor the conditional probability $p(d,q|r)$, corresponding to "document generation" and "query generation."

Using document generation, $p(d,q|r) = p(d|q,r)p(q|r)$, the following ranking formula is obtained:

$$\log \frac{p(r|q,d_j)}{p(\overline{r}|q,d_j)} = \log \frac{p(d_j|q,r)}{p(d_j|q,\overline{r})} + \log \frac{p(r|q)}{p(\overline{r}|q)}. \tag{2.6}$$

Based on document generation, many classical probabilistic retrieval models [49, 114, 145], including the Binary Independence Retrieval (BIR) model [49, 114], have been developed to estimate the probability of relevance.

Let us now consider refining Eq. 2.5 with query generation, i.e., $p(d,q|r) = p(q|d,r)p(d|r)$. In this case, we obtain the following formula:

$$\log \frac{p(r|q,d_j)}{p(\overline{r}|q,d_j)} = \log \frac{p(q|d_j,r)}{p(q|d_j,\overline{r})} + \log \frac{p(r|d_j)}{p(\overline{r}|d_j)}. \tag{2.7}$$

Various models based on query generation have been explored in [49, 74]. Under the assumption $R = \overline{r}$, the document d_j is conditionally independent of the query q, the formula can be transformed to be

$$\log \frac{p(r|q,d_j)}{p(\overline{r}|q,d_j)} \stackrel{rank}{=} \log p(q|d_j,r) + \log \frac{p(r|d_j)}{p(\overline{r}|d_j)}. \tag{2.8}$$

The major component $p(q|d_j, r)$ is the probability that a user would use q as a query to retrieve d_j. The second component $p(r|d_j)$ is a prior which is usually ignored.

Over the decades, an interesting class of probabilistic models called language modeling approaches have led to effective retrieval functions. The language modeling approach was first introduced by Ponte and Croft in [108]. The goal is to infer a language model for each document and rank according to the estimated probability $p(q|d_j)$ of the query given the language model of document d_j. Many variations of the basic language modeling have since been proposed and studied, including relevance-based language model [75], title language model [60], cluster-based language models [84], etc. Typically, a necessary step for these language models is to perform smoothing for the unseen query terms in the document. To improve the accuracy of the estimated model, several different smoothing methods [145, 146, 147, 148], such as Jelinek-Mercer smoothing and Bayesian smoothing using Dirichlet priors, have been proposed which plays a similar role to term weighting in a traditional vector space model.

Language models are attractive because of their foundations in statistical theory. Here we study the basic language modeling approach. To determine the probability of a query given a document, we infer a document language model θ_d for each document. The relevance score of document d with respect to query q is then defined as the conditional probability $p(q|\theta_d)$. Suppose $q = t_1...t_m$ and each term t is generated independently, the relevance score would be,

$$p(q|\theta_d) = \prod_{t \in q} p(t|\theta_d)^{c(t,q)}, \tag{2.9}$$

where $c(t, q)$ is the count of term t in query q, and $p(t|\theta_d)$ is the maximum likelihood estimator of the term in a document d. With such a model, the retrieval problem is reduced to the problem of estimating $p(t_i|\theta_d)$. In general,

the Dirichlet prior smoothing method [148] is employed to assign nonzero probabilities to unseen words as follows:

$$p(t|\theta_d) = \frac{c(t,d) + \nu p(t|C)}{|d| + \nu}. \tag{2.10}$$

where ν is the parameter to control the amount of smoothing, and $p(t|C)$ is the collection language model. As the superior performance achieved by the statistical language model, in this thesis, we employ the statistical language model as the baseline model with the content information for several Web mining applications.

2.2 Web Link Analysis

The analysis of hyperlinks and the graph structure has been extensively studied in the development of Web search and mining. The link analysis methods are basically used for ranking Web search results as one of many factors in computing a composite score for a Web page or document. In this section, we briefly review two fundamental methods, PageRank [18] and HITS [69], as well as some other variations for link analysis.

2.2.1 PageRank

The intuition of PageRank is that a link from page j to page i represents a vote for page i, by page j. The pages which are linked by many "important" pages become more "important". The PageRank of a page is defined recursively and roughly based on the number of inbound links (inlinks) as well as the PageRank of the pages providing the links. Given the Web graph $G = (V, E)$, the PageRank score of the page i (denoted by $P(i)$) is formally defined by

$$P(i) = (1-d)\frac{1}{N} + d \sum_{(j,i)\in E} \frac{P(j)}{O_j}, \tag{2.11}$$

where N is the total number of pages, and O_j is the number of outbound links of page j. The above formula can be interpreted as a random surfer model who gets bored after several clicks and switches to a random page. The parameter d is a damping factor which can be set between 0 and 1, and it is usually set around 0.85. Because of the size of the actual Web, the PageRank values are calculated using an approximatively iterative computation. This means that each page is assigned an initial starting value 1 and the PageRanks of all pages are then calculated in several computation circles based on the above equation. Finally, the sum of all pages' PageRank values still converges to the total number of Web pages.

2.2.2 HITS

HITS (Hyperlink-Induced Topic Search) [69] is another important link analysis algorithm that determines two values for each page, a authority score and a hub score, instead of a single value in PageRank. The intuitive notions behind the HITS algorithm are that good hubs point to good authorities and that good authorities are linked by good hubs.

In the HITS algorithm, authority and hub values are defined in terms of one another in a mutual recursion. To begin the calculation, the authority value $A(i)$ and the hub value $H(i)$ of each page are set to be 1. Then, the iterative algorithm with a pair of updates is given by the follow equations:

$$A(i) \leftarrow \sum_{j \mapsto i} H(j) \tag{2.12}$$

$$H(i) \leftarrow \sum_{i \mapsto j} A(i), \tag{2.13}$$

where $i \mapsto j$ means there exists a link from i to j. Thus, an authority value is computed as the sum of the scaled hub values that point to that page, and meanwhile, a hub value is the sum of the scaled authority values of the pages it points to. Essentially, a good hub represents a page that points to many

other pages, and a good authority represents a page that is linked by many different hubs.

2.2.3 Other Variations

There are a family of variations proposed based on the original PageRank and HITS, such as topic-sensitive PageRank [53, 54], EigenTrust [67], PopRank [101], TrustRank [52], BrowseRank [85], and so on. In addition, a family of work on the structural re-ranking paradigm over a graph is proposed to refine the initial ranking scores based on centrality within graphs, through PageRank-inspired algorithm [72] and HITS-style cluster-based approach [73]. In [43], the authors have tried to model a unified framework for link analysis, which includes HITS and PageRank. Several normalized ranking algorithms are studied which are intermediate between HITS and PageRank.

Beyond explicit hyperlinks on the Web, the PageRank and HITS algorithms can be utilized to explore other implicit links in other contexts. PopRank [101] is developed to extend PageRank models to integrate heterogenous relationships between objects. Another approach suggested by Minkov et al. [98] has been used to improve an initial ranking on graph walks in entity-relation networks. Cohn and Hofmann [28] propose pLSI+PHITS to construct the latent space by combining content with link information, using content analysis based on pLSI [57] and link analysis based on PHITS [27]. Zhang et al. [149] propose a method to improve Web search results based on a linear combination of results from text search and authority ranking. However, the linear combination does not make full use of the information as it treats each of them individually. In this thesis, we propose the Co-HITS algorithm [41], which contains HITS and PageRank as special cases, and it integrates the graph information with the content information simultaneously.

2.3 Semi-supervised Learning

Semi-supervised learning [26, 154] considers the problem of learning from both a set of labeled data and a set of unlabeled data. In recent research studies, many methods have been proposed for solving semi-supervised learning problems, such as co-training [16], self-training [118], transductive support vector machine [63, 130], and a set of graph-based methods [13, 14, 132, 151, 156, 157]. Let us briefly review several graph-based models.

The graph-based semi-supervised learning can be modeled as a random walk with label propagation from labeled data to unlabeled data in [155, 156]. From a different perspective, this method can be viewed as having a quadratic loss function with infinity weight, so that the labeled data are fixed at given label values, and a regularizer based on the graph information:

$$R = \frac{1}{2}\sum_{i,j}^{n} w_{ij}(f_i - f_j)^2 + \sum_{i \in L}(f_i - y_i)^2, \tag{2.14}$$

where w_{ij} corresponds to the weight between point i and point j, L is the set of labeled data, and y_i is the label value. In the equation, the second component only considers the loss function using the labeled data. The local and global consistency method proposed by Zhou et al. [151] uses the loss function based on both labeled and unlabeled data, and the *normalized graph Laplacian* in the regularizer,

$$R = \frac{1}{2}\sum_{i,j}^{n} w_{ij}(\frac{f_i}{\sqrt{D_{ii}}} - \mu\frac{f_j}{\sqrt{D_{jj}}})^2 + \sum_{i}^{n}(f_i - y_i)^2, \tag{2.15}$$

where D is a diagonal matrix with entries $D_{ii} = \sum_j w_{ij}$, and $\mu > 0$ is the regularization parameter. The first term of the right-hand side in the cost function is the smoothness constraint, which means that a good classifying function should not change too much between nearby points. The second term is the fitting constraint, which means a good classifying function should

not change too much from the initial label assignment. The trade-off between these two competing constraints is captured by the parameter μ. By minimizing the cost function R, the solution can be obtained which is equivalent to that of the iterative label propagation algorithm. Motivated by the graph-based semi-supervised learning, we develop the Co-HITS algorithm and graph-based regularization model in Chapter 4 and Chapter 5, respectively.

2.4 Query Log Analysis

With the advance of Web mining technologies, many approaches have been proposed to utilize and analyze query logs to enhance the search results in various aspects. We apply the proposed models to query log analysis in Chapter 3 and Chapter 4.

A common model for utilizing query logs from search engines is in the form of a click graph [31]. Based on the click graph, many research efforts in query log analysis have been devoted to query clustering [11, 139], query suggestion [66, 86, 96], query classification [81, 127, 128, 129] and user behavior understanding [15, 32, 45, 111]. The use of the click-through data for query clustering has been suggested by Beferman and Berger [11], who proposed an agglomerative clustering technique to identify related queries and Web pages. Wen et al. [139] combined query content information and click-through information and applied a density-based method to cluster queries. The click-through data has been studied for query expansion in the past [33, 142]. In addition to query clustering, click-through data has also been used to learn the rank function [64, 110]. Craswell and Szummer [31] used click graph random walks for relevance rank in image search. Mei et al. [96] proposed an approach to query suggestion by computing the hitting time on a click graph. Li et al. [81] presented the use of click graphs in improving query intent classifiers. These methods are proposed based on the click graph, while

the objective of our proposed entropy-biased model [39] is to investigate a better model to utilize and represent the click graph.

In Chapter 3, we propose an entropy-biased framework to find a better representation through modeling click graphs. It is unique in which we focus solely on how to represent query using the click graph. There are several approaches that have tried to model the representation of queries or documents on the click graph. Baeza-Yates et al. [6] used the content of clicked Web pages to define a term-weight vector model for a query. They considered terms in the URLs clicked after a query. Each term was weighted according to the number of occurrences of the query and the number of clicks of the documents in which the term appeared. In [7], the authors introduced another vectorial representation for the queries without considering the content information. Queries were represented as points in a high dimensional space, where each dimension corresponds to a unique URL. The weight assigned to each dimension was equal to the click frequency. This is one of the traditional click frequency models. Moreover, Poblete et al. [106] proposed the query-set document model by mining frequent query patterns to represent documents rather than the content information of the documents. However, these existing methods do not distinguish the variation on different query-URL pairs.

Besides, there is a trend to explore the query logs and model queries with variation for personalization [44, 136]. Dou et al. [44] explored click entropy to measure the variability in click results, while Teevan et al. [136] proposed result entropy to capture how often results change. In Chapter 3, we also utilize the entropy information of the URL. Other methods are focused on personalization for different queries, while our proposed entropy-biased models are different, which are focused on the weighting scheme of various query-URL pairs. Another group of query log analysis aims to explain the log generation process [32, 46] and understand user behavior [45, 111, 140]. Dupret et

al. [46] interpreted the data found in search engine logs by two factors: the position of the document in the result list and the attractiveness of the document surrogate. A fundamental problem in click data is the position bias. Craswell et al. [32] attempted to explain that bias by modeling how probability of click depends on position. The proposed entropy-biased models [39] are not trying to model the position-biased data, but these models seem to be able to diminish the position-bias clicks by the inverse query frequency. Furthermore, to incorporate the bipartite graph with the content information from both sides, we proposes a general Co-HITS algorithm [41] and apply it to the application of query suggestion by mining the query logs in Chapter 4.

2.5 Expertise Retrieval

As mentioned before, the objective of this thesis is to propose a general Web mining framework to combine the content with the graph information as well as other kinds of information effectively. In addition to the application to query log analysis, we also address a high-level expertise retrieval task and investigate several models to combine more heterogeneous information in Chapter 5 and Chapter 6.

With the inclusion of expert finding in the TREC Enterprise track [30, 133], a great deal of work has been done in this area. In general, there are two principal approaches for modeling expertise: the candidate model and the document model [8, 48, 103]. These two models have been proposed and compared by Balog et al. [8]. The candidate-based approach is also referred as profile-based method in [48] or query-independent approach in [103]. These methods build a profile ("virtual document") [10] for each candidate based on all documents associated with the candidate, and estimate the ranking scores according to the candidate profile in response to a given query. On the other hand, document-based models [8, 48] are also referred to as query-dependent

method in [103]. These approaches first rank documents in the corpus for a given query topic, and then find the associated candidates according to the retrieved documents. These two kinds of models have their advantages and disadvantages. In terms of data management, candidate-based methods may require significantly smaller data in size than the original corpus. However, the contribution of each document in a profile cannot be measured individually. Meanwhile, document-based models allow the application of advanced text modeling techniques in ranking individual documents, which achieve better performance than the candidate-based models. We choose the document-based model as the baseline, and propose several methods to further enhance this model with valuable graph and community information.

Based on both the candidate and the document models, an expert-finding system has to discover documents related to a person and estimate the probability of that person being an expert from the text. One of the state-of-the-art approaches is based on statistical language models, which have been studied extensively for information retrieval in recent years [108, 145, 148]. Furthermore, Mimno and McCallum [97] propose an Author-Persona-Topic model for matching papers with reviewers, in which the expertise is modeled by multiple topical mixtures associated with each individual author. Wei and Croft [138] describe a topic model for information retrieval tasks. The authors find that interpolations between Dirichlet smoothed language models and topic models show improvements in retrieval performance above language models. However, all those methods only consider the relevance between a query and documents. In our proposed weighted language models, we take into account not only the relevance between a query and documents, but also the importance of the documents.

Besides the categories described above, there are various methods proposed to extend or enhance the expertise retrieval in many ways. Macdonald

and Ounis present a voting model for expert search in [88], and enhance the expert search with query expansion techniques in [89]. In [87], the authors extend the expert search by identifying some high quality evidence. In [40], the authors propose a graph-based re-ranking model and apply it to expert finding for refining the ranking results. Furthermore, Karimzadehgan et al. [68] leverage the organizational hierarchy to enhance expert finding. Serdyukov et al. [125] model the process of expert finding by the multi-step relevance propagation over the expertise graphs. Nevertheless, our proposed community-aware strategies [38] are different from previous methods. In this thesis, we utilize the AuthorRank [83] to measure the authority based on the coauthorship network [100], but it is independent of any query. We develop the query-sensitive AuthorRank as well as the adaptive ranking refinement strategy for the enhanced model.

Most of the previous work has been concentrated on expertise retrieval in enterprise corpora [8] or intranet dataset [9]. Despite all these tasks in expert finding, little work has been done for expertise search on a specific academic field. In [37], Deng et al. introduce three formal models for expert finding in a real world academic field based on the DBLP bibliography. Li et al. [79] build an academic expertise oriented search service, and they propose a relevancy propagation-based algorithm using the co-authorship network for expert finding. Actually, there are some major differences for finding experts from enterprise corpora to DBLP bibliography data. For the enterprise corpora, many research efforts have concentrated on estimating and capturing the association of a candidate with the documents, while it is easy to build the document-candidate associations, i.e., the paper-author bipartite graph, based on the DBLP bibliography data. One shortcoming of DBLP bibliography data is that the information provided by the title is too limited to represent the paper. To address this problem, we utilize Google Scholar as

data supplementation. We investigate a graph-based regularization technique to refine the relevance scores in Chapter 5, and further enhance the performance of the weighted language model using community-aware strategies in Chapter 6.

□ **End of chapter.**

Chapter 3

Entropy-biased Models for Click Graphs

In this chapter, we investigate and develop a novel entropy-biased framework to find a better query representation in order to better compute the similarity of queries through modeling click graphs. We focus solely on how to represent a query by a vector of documents based on the click graph. The intuition behind this model is that various query-URL pairs should be treated differently, i.e., common clicks on less frequent but more specific URLs are of greater value than common clicks on frequent and general URLs. According to this intuition, we utilize the entropy information of the URLs and introduce a new concept, namely the inverse query frequency (IQF), to weigh the importance of a click on a certain URL. Furthermore, this IQF weighting scheme is incorporated with the click frequency and user frequency information on the click graph for an effective query representation. To illustrate our methodology, we conduct experiments with the AOL query log data for query similarity analysis and query suggestion tasks. Experimental results demonstrate that considerable improvements in performance are obtained with our entropy-biased models.

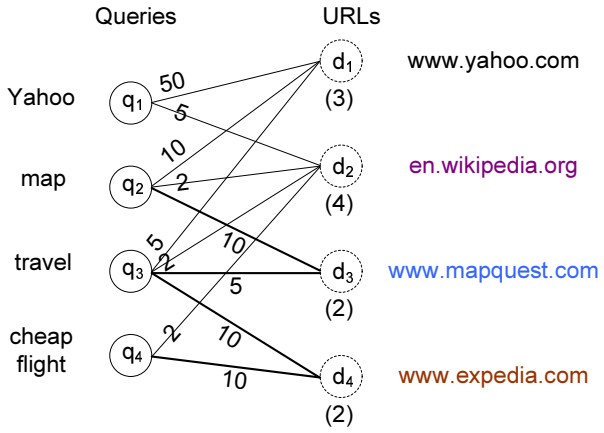


Figure 3.1: Example of a click graph.

3.1 Problem and Motivation

Recently query log analysis has been studied widely for improving search engines' efficacy and usability. Such studies mined the logs to improve numerous search engine's capabilities, such as query suggestion and classification, ranking, targeted advertising, etc. The *click graph* [31], a bipartite graph between queries and URLs, is an important technique for describing the information contained in the query logs, in which edges connect a query with the URLs that were clicked by users as a result. An example of a click graph with 4 queries and 4 URLs is depicted in Figure 3.1. The edges of the graph can capture some semantic relations between queries and URLs. For example, queries "map" and "travel" are related to each other, since they are co-clicked with some URLs such as "www.mapquest.com" and so on. Therefore, how to utilize and model the click graph to represent queries becomes an interesting and challenging problem.

Traditionally, the edge of the click graph is weighted based on the raw *click frequency* (number of clicks) [31] from a query to a URL. The transition probability can be further determined by the normalized click frequency [96, 107].

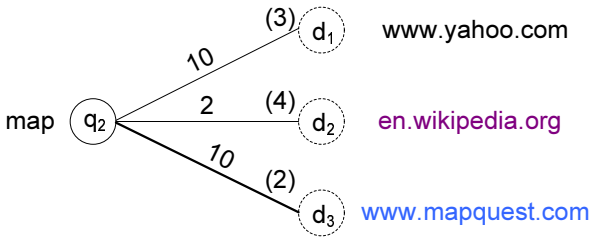


Figure 3.2: The click frequency from the query "map".

Taking the edge from "map" to "www.mapquest.com" in Figure 3.1 as an example, the raw click frequency is 10 and the normalized click frequency is 10/22. However, the traditional query representation for the click graph has its own disadvantages. One of these disadvantages is its robustness, i.e., a query that has a skewed click count on a certain URL may exclusively influence the click graph, such as navigational queries. In order to avoid the adverse effect on learning algorithms, previous work presented in [81] simply identified some navigational queries and removed them from the click graph. Unfortunately, the deletion of such queries leads to the loss of some information. Another related problem is that the raw click frequency can be easily manipulated as it is prone to spam by some malicious clicks. To deal with these critical problems, we explore a novel entropy-biased framework which incorporates raw click frequencies and other information with the entropy information of the connected URLs. Also, there is the issue of an inherent bias of clicks in this graph, favoring already highly ranked URLs [32, 111].

The basic idea of the entropy-biased model is that various query-URL pairs should be treated differently. Let us look at the query "map" (q_2) and its connected URLs, which is shown in Figure 3.2. The click frequency from q_2 to d_3 is the same as the count (10) from q_2 to d_1. There is a critical question when only consider the raw click frequency: *Is a single click on different URLs in the click graph equally important?* Clearly not! In this case, at

an intuitive level, one click on d_3 may capture more meaningful information, or be more important than one click on d_1. The key difference is that the connected URLs are different: One URL is "www.mapquest.com", which is connected with 2 queries; while another URL is "www.yahoo.com", which is connected with 3 queries. Before performing a theoretical analysis, we first briefly review the entropy and information theory [126]. Suppose there is a URL which is commonly clicked and connected with most of the queries (with equal probability), this tends to increase the ambiguity (uncertainty) of the URL. However, if the URL is clicked and connected with fewer queries, this tends to increase the specificity of the URL. A frequently clicked URL thus functions in retrieval as a nonspecific URL, even though its meaning may be quite specific in the ordinary sense. Therefore, *a single click on a specific URL is most likely to be more important for distinguishing the specificity of the query than another click on an ambiguous URL*. Based on the above intuition, we introduce a new concept, denoted as the *inverse query frequency*, to weigh the importance of a click on a certain URL, which can be extended and used for other bipartite graphs.

Consequently, we propose a novel entropy-biased model, namely CF-IQF model, to represent the query, which simultaneously combines the inverse query frequency information with the raw click frequency. As the raw click frequency can be easily manipulated, we develop and use the number of users associated with the query-URL pair, namely the *user frequency* (UF model), instead of the raw click frequency (CF model) to improve the resistance against malicious click data. Moreover, the inverse query frequency can be incorporated with the user frequency, as another entropy-biased UF-IQF model, to achieve better performance.

In a nutshell, our contributions of this chapter are: (1) the introduction of a new notion, namely the *inverse query frequency*, to weigh the importance of

a click on a certain URL, which can be extended and used for other bipartite graphs; (2) the identification of a new source, called the *user frequency*, for diminishing the manipulation of the malicious clicks; (3) the framework of the *entropy-biased model* for the click graph, which simultaneously combines the inverse query frequency with the click frequency and user frequency information; and (4) the *first formal model* to distinguish the variation on different query-URL pairs in the click graph.

The rest of this chapter is organized as follows. Section 3.2 presents the proposed query representation models. Section 3.3 describes two basic applications of these models, which are the query similarity analysis and query suggestion. Section 3.4 describes and reports the experimental evaluation. Section 3.5 summarizes this chapter.

3.2 Query Representation Models

As stated above, the issue of how to represent queries based on the click graph is critical to the task of effectively analyzing query logs. In this section, we first introduce the preliminaries and notations, and then investigate and explore the query representation models for the click graph.

3.2.1 Preliminaries and Notations

Let $Q = \{q_1, q_2, ..., q_M\}$ be the set of M unique queries submitted to a search engine during a specific period of time. Let $D = \{d_1, d_2, ..., d_N\}$ be the set of N URLs clicked for those queries. A *click graph* is a query-URL bipartite graph $G = (Q \cup D, E)$ where every edge in E connects a vertex in the query set Q and one in the URL set D. For $q \in Q$ and $d \in D$, the pair (q, d) is an edge of E if and only if there is a user who clicked on URL d after submitting the query q. For each edge $(q_i, d_j) \in E$, we associate a numeric weight c_{ij}, known as the *click frequency*, that measures the number of times the URL

Table 3.1: Click frequency matrix for the example click graph.

C	d_1	d_2	d_3	d_4
q_1	50	5	0	0
q_2	10	2	10	0
q_3	5	2	5	10
q_4	0	2	0	10

d_j was clicked when shown in response to the query q_i. Let C be an $M \times N$ matrix, whose M rows correspond to the queries of Q and whose N columns correspond to the URLs of D, and the entry (i, j) contains a value c_{ij}. The click frequency matrix of Figure 3.1 is shown in Table 3.1.

Let $U = \{u_1, u_2, ..., u_K\}$ be the set of K users who submitted the queries and clicked on the URLs. Now, a query instance can be made up of one or more $\langle q, d, u\rangle$ triples. It is obvious that every edge (q_i, d_j) in the click graph has a set of users associated with it, so we introduce a new notion uf_{ij}, referred to as the *user frequency*, that measures the total number of users who submitted the query q_i and clicked on the URL d_j. This measurement can be a good supplement of the click frequency for a robust query representation. To further explore the information of query logs, we aggregate the number of queries that are connected with a URL d_j and use $n(d_j)$ to denote it. A URL with large $n(d_j)$ means the document is commonly clicked on many queries, which tends to increase the ambiguity and uncertainty of the URL according to information theory [126]. Therefore, we introduce another novel and important notion $idf(d_j)$, the *inverse query frequency*, to denote the general importance of a certain URL d_j. Some other notations are briefly shown in Table 3.2, and will be defined in the following subsections.

Table 3.2: Table of Notation.

Symbol	Meaning
C	$M \times N$ query-URL matrix
c_{ij}	Click frequency between query q_i and URL d_j, with the entry (i, j) of the matrix C
$cf(q_i)$	Number of clicks for query q_i
$cf(d_j)$	Number of clicks for URL d_j
uf_{ij}	User frequency between q_i and d_j
$n(d_j)$	Number of queries associated with URL d_j
$idf(d_j)$	Importance of a certain URL d_j
$p(d_j\|q_i)$	Transition probability from q_i to d_j
$p(q_i\|d_j)$	Transition probability from d_j to q_i
$p(q_j\|q_i)$	Transition probability from query q_i to q_j
P_{q2d}	An $M \times N$ query-URL probability matrix
P_{d2q}	An $N \times M$ URL-query probability matrix
P_{q2q}	$M \times M$ query-query probability matrix

Table 3.3: CF transition probabilities for the example click graph.

P_{q2d}	d_1	d_2	d_3	d_4
q_1	0.909	0.091	0	0
q_2	0.455	0.091	0.455	0
q_3	0.227	0.091	0.227	0.455
q_4	0	0.167	0	0.833

3.2.2 Click Frequency Model

Traditionally, the edge of the click graph is weighted by the raw click frequency between a query and a URL, which we call *click frequency (CF) model*. Given $q_i \in Q$ and $d_j \in D$, the transition probability [31, 96, 107] from the query q_i to the URL d_j is defined by normalizing the click frequency from the query q_i as

$$p(d_j|q_i) = \frac{c_{ij}}{cf(q_i)}, \tag{3.1}$$

where $cf(q_i) = \sum_{j \in D} c_{ij}$, and it denotes the aggregated number of clicks for q_i. The notation $p(q_i|d_j)$ denotes the transition probability from the URL d_j to the query q_i,

$$p(q_i|d_j) = \frac{c_{ij}}{cf(d_j)}, \tag{3.2}$$

where $cf(d_j) = \sum_{i \in Q} c_{ij}$, and it denotes the aggregated number of clicks for the URL d_j. Although the click frequency c_{ij} is the same, the transition probabilities $p(q_i|d_j)$ and $p(d_j|q_i)$ are generally not symmetric because of the various normalization. If there is no edge between q_i and d_j, the transition probability is equal to 0.

After calculating all these transition probabilities, we obtain two kinds of matrices: $P_{q2d} \in \mathbb{R}^{M \times N}$ and $P_{d2q} \in \mathbb{R}^{N \times M}$. Taking the click graph of Figure 3.1 as an example, we can get the transition matrix P_{q2d} as shown in

Table 3.3. Without considering the content information, the query q_i can be represented by a vector of documents weighted as the i-th row of the matrix P_{q2d}:

$$\overrightarrow{q_i} = \langle P_{q2d}(i,1), ..., P_{q2d}(i,N) \rangle,$$

and meanwhile the document d_j can be represented by a vector of queries weighted as the j-th row of the matrix P_{d2q}:

$$\overrightarrow{d_j} = \langle P_{d2q}(j,1), ..., P_{d2q}(j,M) \rangle.$$

After vectorization, it can be used to measure the similarity between queries and applied to other query log analysis. According to Table 3.3, for example, the most similar query of q_2 ("map") is q_1 ("Yahoo") using the cosine similarity.

3.2.3 Entropy-biased Model

The CF model only considers the raw click frequency, and treats different query-URL pairs equally even if some URLs are very heavily clicked. More generally, a great variation in URL distribution is likely to appear, and it may thus cause the loss of important information since different query-URL pairs are not sufficiently distinguished. For example, the click frequency c_{21} is equal to c_{23} in Figure 3.1. However, it may be more reasonable to weight these two edges differently because of the variation of the connected URLs.

In this chapter, we define $int(q,d)$ to be *true* when the query q has clicks on d at least once. Let $n(d_j)$ be the total number of queries (*query frequency*) that are connected with the URL d_j, which is defined as

$$n(d_j) = \sum_{i \in Q} 1_{int(q_i, d_j)}.$$

It is predicted that the more general and highly ranked URL would be clicked and connected with more queries than the specific URLs. Thus the less specific URLs would have a larger collection distribution than the more specific

ones, which tends to increase the ambiguity and uncertainty of the URLs in the ordinary sense. Using information theory, the entropy [126] of a URL d_j is defined as

$$E(d_j) = -\sum_{i \in Q} p(q_i|d_j) \log p(q_i|d_j). \tag{3.3}$$

Suppose that the URL d_j is connected with those queries with equal probability $p(q_i|d_j) = \frac{1}{n(d_j)}$, the maximum entropy is transformed to

$$E(d_j) = \log n(d_j). \tag{3.4}$$

Generally, the entropy of the URL tends to be proportional to the query frequency $n(d_j)$. In order to simplify the calculation, we roughly use the maximum entropy to approximate the exact entropy in the following analysis.

It is argued that the discriminative ability of a URL should be inversely proportional to the entropy, hence a (heavily-clicked) URL with a high query frequency is less discriminative overall. This motivates us to propose a novel and important concept, referred to as the *inverse query frequency*, to measure the discriminative ability of the URL d_j. Suppose $|Q|$ is the total number of queries in the query log, the *inverse query frequency* for the URL d_j is defined as,

$$iqf(d_j) = \log |Q| - \log n(d_j) = \log \frac{|Q|}{n(d_j)}, \tag{3.5}$$

which is similar to the inverse document frequency for the term [65]. Table 3.4 shows the corresponding IQF values of the example URLs. The inverse query frequency factor has several benefits. The most important one is that it can constrain and diminish the influence of some heavily-clicked URLs. This will tend to balance the inherent bias of clicks for those highly ranked URLs [32]. Furthermore, the inverse query frequency can be incorporated with other factors to tune the representation models as shown in the following subsections.

Table 3.4: IQF values of the URLs

	d_1	d_2	d_3	d_4
iqf	log(4/3)	0	log(2)	log(2)

CF-IQF Model

In the entropy-biased model, we incorporate the inverse query frequency with the raw click frequency in a unified *CF-IQF model*, namely

$$cfiqf(q_i, d_j) = c_{ij} \cdot iqf(d_j). \tag{3.6}$$

The intuition behind the CF-IQF model is that query-URL pairs are treated differently according to the inverse query frequency, so that the common clicks on less frequent yet more specific URLs are of greater value than the common clicks on frequent URLs. Figure 3.3 shows the surface specified by the click frequency, query frequency, $cfiqf$, with color specified by the $cfiqf$ value. The color is proportional to the surface height. A high weight $cfiqf$ is reached by a high click frequency for the query-URL pair and a low query frequency associated with the URL in the whole query log. As shown in Figure 3.3, the query-URL pair A, which has the same click frequency with B, will be weighted much higher than B because of the associated inverse query frequency, hence such weights tend to diminish the influence of heavily-clicked URLs.

The new transition probability from q_i to d_j becomes

$$p'_c(d_j|q_i) = \frac{cfiqf(q_i, d_j)}{cfiqf(q_i)}, \tag{3.7}$$

where $cfiqf(q_i) = \sum_{j \in D} cfiqf(q_i, d_j)$. The new matrix P'_{q2d} of Figure 3.1 is shown in Table 3.5. Based on this matrix, it can be calculated that the most similar query of $q2$ ("map") is $q3$ ("travel"), which is more reasonable than

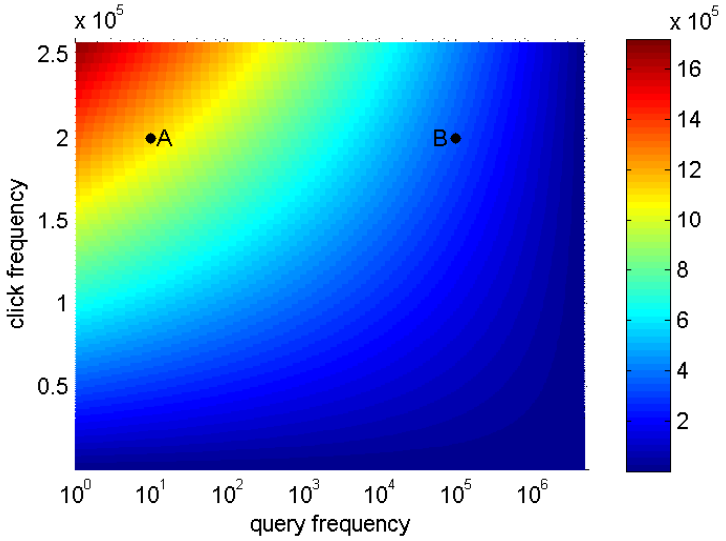


Figure 3.3: The surface specified by the click frequency, query frequency and cfiqf, with color specified by the cfiqf value. The color is proportional to the surface height.

the result of CF model. Currently, we only consider changing the transition probability from the query to the URL, and keeping the transition probability $p(q_i|d_j)$ from the URL to the query as the same as that of CF model.

UF Model and UF-IQF Model

Another drawback of the CF model is that it is prone to spam by some malicious clicks, and it can be easily influenced by a single user if he/she clicked on a certain URL thousands of times. To address the problem, we introduce a new concept *user frequency* (UF), which denotes the number of users associated with the query-URL pair, instead of the click frequency, to improve the resistance against malicious click data. Let $int(q_i, d_j, u_k)$ to be *true* if a user u_k submitted the query q_i and clicked on the URL d_j at least

Table 3.5: CF-IQF transition probabilities for the example click graph.

P'_{q2d}	d_1	d_2	d_3	d_4
q_1	1	0	0	0
q_2	0.293	0	0.707	0
q_3	0.122	0	0.293	0.586
q_4	0	0	0	1

once, then the user frequency uf_{ij} is defined as

$$u_{ij} = \sum_{k \in U} 1_{int(q_i, d_j, u_k)}.$$

Based on the user frequency, we can obtain *UF model* similar to CF model. Intuitively, UF model reinforces the capability of diminishing the effect of some manipulated clicks.

To further distinguish the performance of the model, we also incorporate the user frequency with the inverse query frequency in a unified *UF-IQF model*,

$$ufiqf(q_i, d_j) = uf_{ij} \cdot iqf(d_j). \tag{3.8}$$

With Eq. 3.8, the transition probability from q_i to d_j becomes

$$p'_u(d_j|q_i) = \frac{ufiqf(q_i, d_j)}{ufiqf(q_i)}, \tag{3.9}$$

where $ufiqf(q_i) = \sum_{j \in D} ufiqf(q_i, d_j)$.

3.2.4 Connection with Other Methods

In this subsection, we establish the connection between our entropy-biased model and the famous TF-IDF model [65, 121]. Over the years, the weighting scheme TF-IDF has been extensively and successfully used in the vector

space model for text retrieval. Several researchers [113, 117, 35] have tried to interpret IDF based on binary independence retrieval, Poisson, information entropy and language modeling. Although the success of the TF-IDF in the text mining is widely claimed, it has never been explored to bipartite graphs. The idea of measuring the discriminative ability of the URL by IQF is totally new, and it can be expected to produce the similar effects on click graphs as IDF on text mining. Moreover, our entropy-biased model is employed to identify the edge weighting of the click graph, which can also be applied to other bipartite graphs without the content information. As the query can also be represented by the vector of terms using TF and TF-IDF models, we will compare the performance of these two models with our proposed models in Section 3.4.3.

3.3 Mining Query Log on Click Graph

The proposed query representation models can be applied to mine the query log in many cases, such as query-to-query similarity, query clustering, query suggestion, etc. For the comparison of different models, we focus on two tasks: (1) the fundamental query-to-query similarity analysis, which is very suitable for evaluating the performance of the proposed query representation models, and (2) the popular query suggestion task, which is to find semantically related queries for a given query using the graph-based random walk model.

3.3.1 Query-to-Query Similarity Measurement

As the query can be represented by a vector of documents (or a vector of terms), two common similarity measurements will be used to calculate the similarity between queries: one is the cosine similarity and the other is the Jaccard coefficient. The cosine similarity is a measure of similarity between

two vectors by finding the angle θ between them. It is represented using a dot product and magnitude as

$$Cos(\theta) = \frac{\overrightarrow{q_i} \cdot \overrightarrow{q_j}}{\| \overrightarrow{q_i} \| \| \overrightarrow{q_j} \|}, \tag{3.10}$$

where $\overrightarrow{q_i}$ denotes the vector of a query. The Jaccard coefficient is defined as the value of the intersection divided by the value of the union of the query vectors:

$$J(\overrightarrow{q_i}, \overrightarrow{q_j}) = \frac{\sum_{n \in N} |P_{q2d}(i,n) \cap P_{q2d}(j,n)|}{\sum_{n \in N} |P_{q2d}(i,n) \cup P_{q2d}(j,n)|}, \tag{3.11}$$

where $P_{q2d}(i,n)$ denotes the n-th value of $\overrightarrow{q_i}$. We report and analyze the query similarity results in Section 3.4.3.

3.3.2 Graph-based Random Walk Model

In previous studies [31, 96, 107], the click graph has been thought of as a random walk between queries and URLs according to the transition probabilities P_{q2d} and P_{d2q}. To consider the vertices in one side, such as the query-to-query graph, then a new random walk can be introduced by the transition probability from q_i to q_j,

$$p(q_j|q_i) = \sum_{k \in D} p(d_k|q_i) p(q_j|d_k). \tag{3.12}$$

We use P_{q2q} to denote the transition matrix whose entry (i,j) has the value $p(q_j|q_i)$. It is important to note that the self-transition probability exists naturally in the model.

The personalized PageRank [54, 59] is the steady-state distribution of the random walk, which is usually used to rank vertices on the graph in a query dependent way. The corresponding linear system of personalized PageRank can be shown as:

$$R_j^{n+1} = (1-\alpha) R_j^{(0)} + \alpha \cdot \sum_i p(q_j|q_i) R_i^n, \tag{3.13}$$

where $R_j^{(0)}$ is a personalized (or query dependent) initial values for vertex j, and n is the steps of a random walk. We may set $R_j^{(0)} = 1$ if v_j is the given query and 0 otherwise. The parameter α is usually set to be 0.7 in previous studies. Since the objective is to show the effectiveness of our proposed models for query suggestion, we present the query suggestions ranked by personalized PageRank in Section 3.4.4.

3.4 Experimental Evaluation

In the following experiments we compare our proposed models with other methods on the tasks of mining query logs through an empirical evaluation. We define the following task: Given a query and a click graph, the system has to identify a list of queries which are most similar or semantically relevant to the given query. In the rest of this section, we introduce the data collection, the assessments and evaluation metrics, and present the evaluation results.

3.4.1 Data Collection and Analysis

The dataset that we study is adapted from the query log of AOL search engine [102]. The entire collection consists of $19,442,629$ user click-through records. These records contain $10,154,742$ unique queries and $1,632,789$ unique URLs submitted from about $650,000$ users over three months (from March to May 2006). As shown in Table 3.6, each record of the click contains the same information: UserID, Query, Rank and ClickURL (we do not show the Time properties due to the limited space). This dataset is the raw data recorded by the search engine, and contains a lot of noises. Hence, we conduct a similar method employed in [137] to clean the raw data. We clean the data by removing the queries that appear less than 2 times, and by combining the near-duplicated queries which have the same terms without the stopwords and punctuation marks (for example, “google’s image” and “google image”

Table 3.6: Samples of the AOL query log dataset.

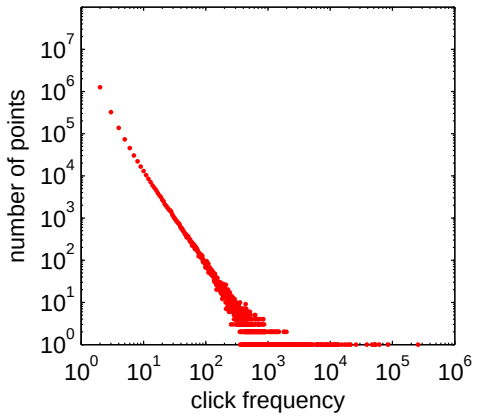
UserID	Query	Rank	ClickURL
2722	yahoo	1	www.yahoo.com
121537	map	1	www.mapquest.com
123557	travel	2	www.expedia.com
1903540	cheap flight	1	www.cheapflights.com

will be combined as the same query). After cleaning, we get totally $883,913$ queries and $967,174$ URLs in our data collection. After the construction of the click graph, we observe that a total of $4,900,387$ edges exist, which indicates that each query has 5.54 distinct clicks, and each URL is clicked by 5.07 distinct queries. Moreover, taken as a whole, this data collection has $250,127$ unique terms which appear in all the queries.

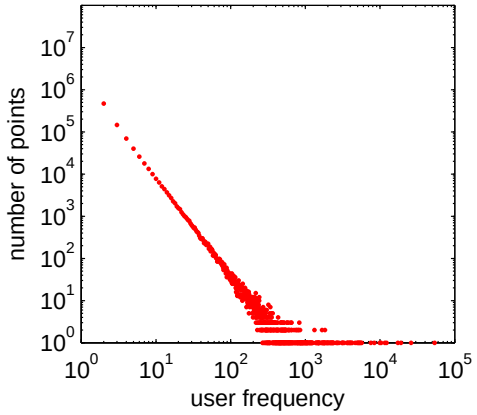
It has been shown in [7] that the occurrences of queries and the clicks of URLs exhibit a power-law distribution. However, the properties of the user frequency and query frequency have not been well explored. Figure 3.4 shows the distributions of the click frequency (c_{ij}) and the user frequency (uf_{ij}) associated with the query-URL edges, and the query frequency ($n(d_j)$) associated with the URLs. All of them exhibit power-law distributions in the figure.

3.4.2 Assessments and Evaluation Metrics

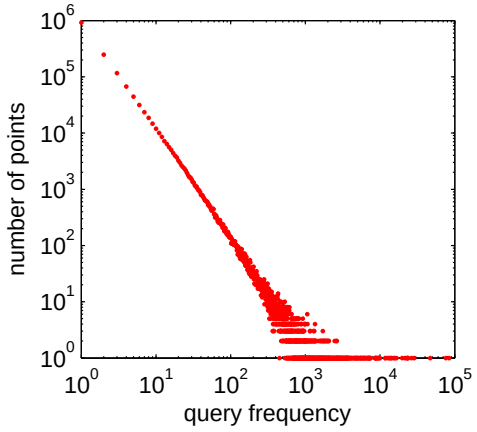
It is difficult to evaluate the quality of query similarity/relevance rankings due to the scarcity of data that can be examined publicly. For an automatic evaluation, we utilize the same method used in [7] to evaluate the similarity



(a)



(b)



(c)

Figure 3.4: The distributions of the (a) click frequency, (b) user frequency and (c) query frequency.

of retrieved queries, but engage the Google Directory[1] instead of the Open Directory Project[2]. When a user types a query in Google Directory, besides site matches, we can also find *category* matches in the form of paths between directories. Moreover, these categories are ordered by relevance. For instance, the query "United States" would provide the hierarchical category "Regional > North America > United States", while one of the results for "National Parks" would be "Regional > North America > United States > Travel and Tourism > National Parks and Monuments". Hence, to measure how similar two queries are, we can use a notion of similarity between the corresponding categories provided by the search results of Google Directory. In particular, we measure the similarity between two categories Ca_i and Ca_r as the length of their longest common prefix $P(Ca_i, Ca_r)$ divided by the length of the longest path between Ca_i and Ca_r. More precisely, the similarity is defined as:

$$Sim(Ca_i, Ca_r) = |P(Ca_i, Ca_r)| / \max(|Ca_i|, |Ca_r|), \tag{3.14}$$

where $|Ca_i|$ denotes the length of a path. For instance, the similarity between the above two queries is 3/5 since they share the path "Regional > North America > United States" and the longest one is made of five directories. We evaluate the similarity between two queries by measuring the similarity between the aggregated categories of the two queries, among the top 5 answers provided by Google Directory.

To give a fair assessment, we randomly select 300 distinct queries from the data collection, then retrieve a list of similar queries using the proposed methods for each of these queries. For the evaluation of the task, we adopt the precision at rank n to measure the relevance of the top n results of the retrieved list with respect to a given query q_r, which is defined as

$$P@n = \frac{\sum_{i=1}^{n} Sim(q_i, q_r)}{n}, \tag{3.15}$$

[1] http://directory.google.com/

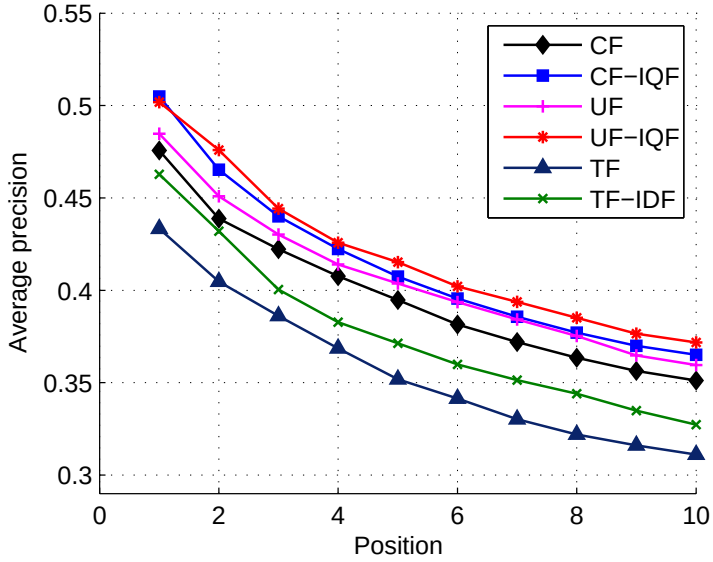
[2] http://www.dmoz.org/

where $Sim(q_i, q_r)$ means the similarity between q_i and q_r. In our experiments, we report the precision from $P@1$ to $P@10$, and take the average over all the 300 distinct queries.

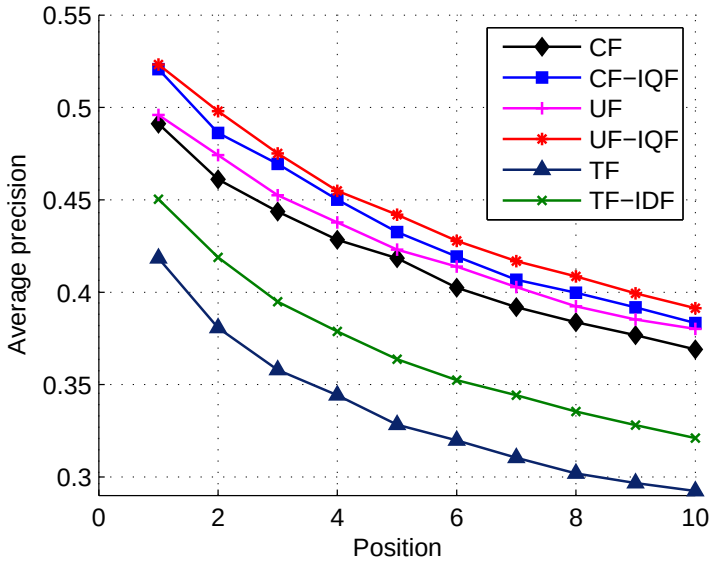
3.4.3 Query Similarity Analysis

We consider the question whether our proposed method can boost the performance using the entropy-biased models for the fundamental query similarity analysis tasks. We compare six different models, including four models (CF, CF-IQF, UF and UF-IQF) based on the click graph and two models (TF and TF-IDF) based on the query content information, and report the precisions from $P@1$ to $P@10$ in Figure 3.5 using two similarity measurements. In this figure we can see, as expected, that our proposed entropy-biased CF-IQF model outperforms the CF model in all the metrics from $P@1$ to $P@10$. Similarly to what happens between the CF-IQF and CF models, the performance of the UF-IQF model is better than that of the UF model. The results support our intuition of the entropy-biased framework about treating various query-URL pairs differently. When comparing the results of UF with CF, and the results of UF-IQF with CF-IQF, we can observe that the UF and UF-IQF models perform better than the CF and CF-IQF models respectively, which indicates the user frequency associated with the query-URL pair is more robust than the click frequency for modeling the click graph.

We also compare our models with the TF and TF-IDF models to see whether the improvements of CF-IQF and UF-IQF over CF and UF models are consistent with the improvement of the TF-IDF over TF model. According to Figure 3.5, it is obvious that the TF-IDF model improves the performance of the TF model, with the same observations of our entropy-biased models. The reason is that they share the same key point to identify and tune the importance of a term or a query-URL edge. The major difference is that



(a) Cosine similarity



(b) Jaccard coefficient

Figure 3.5: The performance comparison of six models (CF, CF-IQF, UF, UF-IQF, TF and TF-IDF models) using two different similarity measurements.

Table 3.7: Comparison of different methods by P@1 and P@10. We also show the percentage of relative improvement in the lower part.

Method	Cosine		Jaccard	
	P@1	P@10	P@1	P@10
CF	0.476	0.351	0.491	0.369
CF-IQF	**0.505**	0.365	0.521	0.383
UF	0.485	0.360	0.500	0.380
UF-IQF	0.502	**0.372**	**0.523**	**0.391**
TF	0.433	0.311	0.418	0.292
TF-IDF	0.463	0.327	0.450	0.321
CF-IQF/CF	**6.12**%	3.96%	6.01%	3.84%
UF-IQF/UF	3.52%	3.38%	**5.50**%	2.92%
UF-IQF/CF	5.49%	5.86%	6.51%	6.01%
TF-IDF/TF	6.78%	5.21%	7.63%	9.79%
CF/TF	9.76%	12.91%	17.41%	26.23%
UF/TF	11.85%	15.61%	18.53%	**30.02**%
CF-IQF/TF-IDF	9.09%	11.57%	15.65%	19.39%
UF-IQF/TF-IDF	8.44%	13.61%	16.19%	**21.89**%

the TF-IDF model is used to find the weight value of a term in a document, which has a significant effect in the information retrieval field. However, our entropy-biased models are applicable in identifying the weight of the edge for the click graph, which can be extended to other bipartite graphs without the content information.

To gain a better insight into the details of the results, we show the comparison of different models using $P@1$ and $P@10$ in Table 3.7. The first part shows the absolute precisions of those models, and the second part illustrates the percentage of relative improvements. A quick scan of the first part, accompanying with Figure 3.5, reveals that UF-IQF achieves the best performance in most cases. When looking at the relative improvements of those models (the top four lines of the lower part), we can see that CF-IQF improves over CF by up to 6.12%, UF-IQF over UF by up to 5.5%, and UF-IQF over CF by up to 6.51%. While TF-IDF improves over TF by up to 9.79% for $P@10$ using Jaccard coefficient, this is because the precision of TF is much lower than other methods, which can be easily be improved. In terms of the final four lines in Table 3.7, another interesting comparison is seen between the proposed models on the click graph and the traditional models on the query content information. Based on the click graph, CF and UF models improve the traditional TF model significantly from 9.76% to 30.02%, while CF-IQF and UF-IQF models also improve the traditional TF-IDF model from 8.44% to 21.89%. The results reconfirm many previous studies [7, 111] that the click graph catches more semantic relations between queries than the query terms. According to the experimental results, we can argue that it is very essential and promising to consider the entropy-biased models for the click graph.

To test the sensitivity of the similarity measurement of our entropy-biased models, we compare the results of the Jaccard coefficient, and find that the improvements are consistent with the cosine similarity, which indicates that

our entropy-biased models are independent of the similarity measurements. In addition, we notice that Jaccard coefficient performs better than cosine similarity using CF, CF-IQF, UF and UF-IQF models on the click graph, while cosine similarity is better than Jaccard coefficient using TF and TF-IDF models on the query content information.

3.4.4 Random Walk Evaluation

In this subsection, we present the comparison of suggestions generated using the same random walk method with CF and CF-IQF models (we do not show the comparison of UF and UF-IQF models due to space constraints and similar results). To better understand the improvements of our entropy-biased models, we evaluate the performance of our methods with different number of steps (from 2 to 50). Figure 3.6 illustrates the precisions ($P@10$) of CF and CF-IQF models for different parameter n. With the increase of n, both models improve their performance, which can also converge quickly after about 10 steps. As shown in Figure 3.6, it is very clear that the CF-IQF model always performs better than the CF model.

We selectively show the detailed results ranked by the transition probabilities in Table 3.8. In general, the top-4 suggestions generated by the CF model and the CF-IQF model are similar, and mostly semantically relevant to the original query. For the first example in Table 3.8, these two models generate the same suggestions, since the transition probabilities in both models are usually similar. From these suggested results, we see that our models not only capture the most common sense, the "american airline", they also successfully predict infrequent query "alcoholics anonymous" as suggestion. After looking into the last two examples, one important observation is that our CF-IQF model can boost more relevant queries as suggestion and reduce some irrelevant queries. To see the suggestions for "east texas real estate",

Table 3.8: Examples of query suggestions generated by two different models on click graph.

CF model	CF-IQF model
Query = aa	
american airlines	american airlines
alcoholics anonymous	alcoholics anonymous
aa.com	aa.com
airlines	airlines
Query = east texas real estate	
google	east texas acreage
east texas acreage	tyler real estate
texas real estate	tyler texas realtors
tyler real estate	texas real estate
Query = home gym equipment	
home gyms	home gyms
gym equipment	gym equipment
treadmills	treadmills
buy.com	edge 329 upright exercise bike

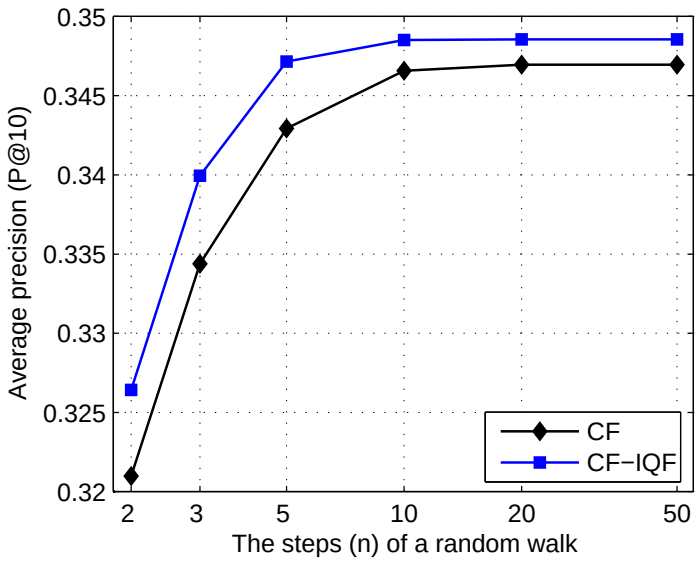


Figure 3.6: The performance of random walk model.

for example, we notice that the first suggestion "google", provided by the CF model, is irrelevant to the original query. This is because there is a edge between the query "east texas real estate" and a heavily-clicked URL "www.google.com", which are highly associated with the query "google" so as to generate the high transition probability from "east texas real estate" to "google". In the last example, the irrelevant suggestion "buy.com" in the CF model arises from the similar reason. Comparing with the CF model, the CF-IQF model can successfully constrain such irrelevant queries and return mostly relevant suggestions (e.g., upright exercise bike), because it reduces the adverse factor in such situations by considering the inverse query frequency in the click graph.

3.5 Summary

In this chapter we present the novel entropy-biased framework for modeling click graphs, whose basic idea is to treat various query-URL pairs differently

according to the inverse query frequency. Although its fundamental concept is very simple, the IQF weighting scheme is never explicitly explored or statistically examined for any bipartite graphs in the information retrieval literature. We not only formally define and quantify this scheme, but also propose the new entropy-biased framework to incorporate it on the click graph for an effective query representation.

To illustrate our methodology, we apply the entropy-biased models to query similarity analysis and query suggestion tasks using the real-world AOL query log data. The main concern is to increase the precision of the top-n retrieved results. For the query similarity analysis, we compare six different models, including four models (CF, CF-IQF, UF and UF-IQF) based on the click graph and two models (TF and TF-IDF) based on the query terms. It is shown that CF-IQF model improves over CF model by up to 6.12%, while UF-IQF over UF by up to 5.5%. As expected, UF-IQF and UF outperform CF-IQF and CF respectively. In addition, UF-IQF model significantly improves the traditional TF-IDF model by up to 21.89%. For the query suggestion task, evaluation results also show that the entropy-biased models outperform the baseline models, indicating that the improvements in our proposed models are consistent and promising. In addition, our method can also be applied to other bipartite graphs. In future work, it would be interesting to apply this entropy-biased model to identify some noise click data.

□ **End of chapter.**

Chapter 4

Generalized Co-HITS Algorithm

In previous chapter, the entropy-biased models aim to find a better query representation through modeling click graphs when only considering the graph information. However, besides the graph information, we could obtain the content information for each entity. As the content and the graph provide different information, it is reasonable to incorporate the content with graph information in a unified framework. In this chapter, we propose a novel and general Co-HITS algorithm to incorporate the bipartite graph with the content information from both sides as well as the constraints of relevance. Moreover, we investigate the algorithm based on two frameworks, including the iterative and the regularization frameworks, and illustrate the generalized Co-HITS algorithm from different views. For the iterative framework, it contains HITS and personalized PageRank as special cases. In the regularization framework, we successfully build a connection with HITS, and develop a new cost function to consider the direct relationship between two entity sets, which leads to a significant improvement over the baseline method.

4.1 Problem and Motivation

Bipartite graphs have been widely used to represent the relationship between two sets of entities (which we refer to as two kinds of data to avoid ambiguity) for Web search and data mining applications. The Web offers rich relational data which can be represented by bipartite graphs, such as queries and URLs in query logs, authors and papers in scientific literature, and reviewers and movies in a movie recommender system. Taking the query-URL bipartite graph as an example, although there is no direct edges between two queries, the edges of the bipartite graph between queries and URLs may lead to hidden edges within the query set as shown in Figure 4.1. Previous work [31] shows that there is a natural random walk on the bipartite graph, which demonstrates certain advantages comparing with the traditional approaches based on the content information. Many link analysis methods have been proposed, such as HITS [69] and PageRank [18], to capture some semantic relations within the bipartite graph.

The problem we address is how to utilize and leverage both the graph and content information, so as to improve the precision of retrieved entities. One good example is the query suggestion by mining a query log, in which we have a query-URL bipartite graph, and the queries and URLs. In addition, the queries and URLs can be represented as term vectors with the content information. The objective of the query suggestion is to find semantically similar queries for the given query q. Traditionally, we can identify initial similar queries based on the content information, then utilize HITS or personalized PageRank [54] for further mutual reinforcement on the bipartite graph. However, one of the issues is that there is a lack of constraints to make sure the final relevance of the score propagation on the graph, as there are many noisy edges within the bipartite graph. For example, let us consider the following two queries: *map* and *Yahoo*, where they may be co-linked by some URLs

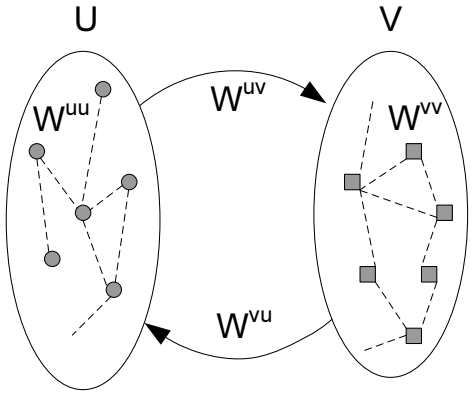


Figure 4.1: Example of a bipartite graph. The edges between U and V are represented as the transition matrices W^{uv} and W^{vu}. Note that the dashed lines represent hidden links when considering the vertices in one side, where W^{uu} and W^{vv} denote the hidden transition matrices within U and V respectively.

such as "www.yahoo.com" (*Yahoo!*). As the general URL *Yahoo!* is associated with many queries, it can aggregate large relevance scores by the mutual reinforcement, which may propagate the score to the highly connected query *Yahoo* and lead to the high relevance score between *map* and *Yahoo*. In this case, if we consider the content information of the URL *Yahoo!*, the relevance score of the URL *Yahoo!* against the query *map* will be very low. Thus, when incorporating the low relevance of the URL into the mutual reinforcement on the bipartite graph, the final relevance score between *map* and *Yahoo* would be constrained to a lower, but more reasonable score. In order to avoid the adverse effect of noisy data, we argue that the initial relevance scores, from both sides of the bipartite graph, provide valuable and reinforced information as well as the constraints of relevance, which should all be incorporated in a unified framework.

In this chapter, we propose a novel and general algorithm, namely generalized Co-HITS, to incorporate the bipartite graph with the content in-

formation from both sides. Consequently, we investigate the following two frameworks, i.e., iterative framework and regularization framework, for the generalized Co-HITS algorithm from different views. The basic idea of the iterative framework is to propagate the scores on the bipartite graph via an iterative process with the constraints from both sides. The iterative framework contains HITS, personalized PageRank, and the one-step propagation algorithm as the special cases. Furthermore, we develop a joint regularization framework instead of the above iterative algorithm. In the regularization framework, we successfully build the connection with HITS, and develop a new cost function to consider the direct relationship between two entity sets, which leads to a significant improvement over the baseline method. To illustrate our methodology, we apply the generalized Co-HITS algorithm with different settings to the query suggestion task using the real-world AOL query log data [102]. Experimental results show that the CoRegu-0.5 (i.e., a model of the regularization framework) achieves the best performance, and its improvements are consistent and promising.

In a nutshell, our major contributions of this chapter are: (1) the introduction of the generalized *Co-HITS* algorithm to incorporate the bipartite graph with the content information from both sides; (2) the investigation of two frameworks, including the iterative and the regularization frameworks, for the generalized Co-HITS algorithm from different perspectives; and (3) a new smoothness function in the regularization framework to consider the direct relationship between two entity sets as well as the smoothness within the same entity set, which leads to a significant improvement over the baseline method.

The rest of this chapter is organized as follows. Section 4.2 briefly introduces the preliminaries. Section 4.3 presents the proposed Co-HITS algorithm, including the iterative framework and the regularization framework.

Section 4.4 describes the application to bipartite graphs. Section 4.5 then reports the experimental evaluation. Some related work and discussions are presented in Section 4.6. Finally, Section 4.7 summarizes this chapter.

4.2 Preliminaries

Consider a bipartite graph $G = (U \cup V, E)$, its *vertices* can be divided into two disjoint sets U and V such that each *edge* in E connects a vertex in U and one in V; that is, there is no edge between two vertices in the same set. Let $U = \{u_1, u_2, ..., u_m\}$ and $V = \{v_1, v_2, ..., v_n\}$ be the two sets of m and n unique entities. Generally, a bipartite graph can be modeled as a weighted directed graph. Given $i \in U$ and $j \in V$, if there is an edge connecting u_i and v_j, the transition probabilities w_{ij}^{uv} and w_{ji}^{vu} are positive, where w_{ij}^{uv} denotes the transition probability from u_i to v_j, and w_{ji}^{vu} denotes the transition probability from v_j to u_i; otherwise, $w_{ij}^{uv} = w_{ji}^{vu} = 0$. Since the transition probability from state i to some state must be 1, we have $\sum_{j \in V} w_{ij}^{uv} = 1$ and $\sum_{i \in U} w_{ji}^{vu} = 1$.

For a bipartite graph, there is a natural random walk on the graph with the transition probability as shown in Figure 4.1. Let $W^{uv} \in \mathbb{R}^{m \times n}$ denote the transition matrix from U to V, whose entry (i, j) contains a weight w_{ij}^{uv} from u_i to v_j. Let $W^{vu} \in \mathbb{R}^{n \times m}$ be the transition matrix from V to U, whose entry (j, i) contains a weight w_{ji}^{vu} from v_j to u_i. To consider the vertices in one side, such as the query-to-query graph in query logs, then a hidden transition probability w_{ij}^{uu} from u_i to u_j, corresponding to a dashed line in Figure 4.1, can be introduced as:

$$w_{ij}^{uu} = \sum_{k \in V} w_{ik}^{uv} w_{kj}^{vu}, \tag{4.1}$$

and

$$\sum_{j \in U} w_{ij}^{uu} = \sum_{j \in U} \sum_{k \in V} w_{ik}^{uv} w_{kj}^{vu} = \sum_{k \in V} \left(w_{ik}^{uv} \sum_{j \in U} w_{kj}^{vu} \right) = \sum_{k \in V} w_{ik}^{uv} = 1. \tag{4.2}$$

Similarly, for the transition probability from v_i to v_j, we can show that $w_{ij}^{vv} = \sum_{k \in U} w_{ik}^{vu} w_{kj}^{uv}$ and $\sum_{j \in V} w_{ij}^{vv} = 1$. We use $W^{uu} \in \mathbb{R}^{m \times m}$ and $W^{vv} \in \mathbb{R}^{n \times n}$ to denote the hidden transition matrices within U and V, respectively.

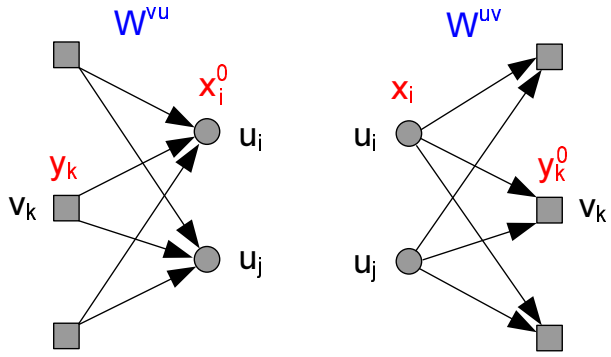
In addition to the graph information, each entity (such as a query or a document) may be represented as a term vector with its content information. For a given query q, the relevance scores of the entities can be calculated using a text relevance function f, such as the vector space model [5] and the statistical language model [108, 147]. The initial relevance scores x_i^0 and y_j^0 are respectively defined by $x_i^0 = f(q, u_i)$, and $y_j^0 = f(q, v_j)$ for u_i and v_j.

4.3 Generalized Co-HITS Algorithm

Given a query q and the above information, the ultimate goal is to find a set of entities which are most relevant to the query q. The problem we address is how to utilize and leverage both the graph and content information, so as to improve the precision of the results. In this section, we propose a novel and general algorithm, namely generalized Co-HITS, to incorporate the bipartite graph with the content information from both sides.

4.3.1 Iterative Framework

The basic idea of our method is to propagate the scores on the bipartite graph via an iterative process. As shown in Figure 4.2(a), the score y_k of v_k is propagated to u_i according to the transition probability. Similarly, additional scores are propagated from other vertices of V to u_i, then the score of u_i is updated to get a new value x_i. In Figure 4.2(b), it shows that the new value x_i is propagated to v_k. The intuition behind the score propagation is the mutual reinforcement to boost co-linked entities on the bipartite graph. In addition, the initial relevance scores based on the content information provide invaluable information, which should also be considered in the framework.



(a) (b)

Figure 4.2: Score propagation on the bipartite graph: (a) score y_k is propagated to u_i and u_j, and (b) score x_i is propagated to v_k.

In order to incorporate the bipartite graph with the content information, the generalized Co-HITS equations can be written as

$$x_i = (1-\lambda_u)x_i^0 + \lambda_u \sum_{k\in V} w_{ki}^{vu} y_k, \tag{4.3}$$

$$y_k = (1-\lambda_v)y_k^0 + \lambda_v \sum_{j\in U} w_{jk}^{uv} x_j, \tag{4.4}$$

where $\lambda_u \in [0,1]$ and $\lambda_v \in [0,1]$ are the personalized parameters, x_i^0 and y_k^0 are the initial scores for u_i and v_k respectively. In this model, the initial scores are normalized to be $\sum_{i\in U} x_i^0 = 1$ and $\sum_{k\in V} y_k^0 = 1$. Thus, after the updating operation, the sum of x_i and the sum of y_k will also be equal to 1 without further normalization. If only considering the vertices in one side, by substituting Eq. (4.4) for y_k in Eq. (4.3), the generalized Co-HITS equation can be represented as the following

$$\begin{aligned} x_i &= (1-\lambda_u)x_i^0 + \lambda_u(1-\lambda_v)\sum_{k\in V} w_{ki}^{vu} y_k^0 + \lambda_u\lambda_v \sum_{j\in U}\left(\sum_{k\in V} w_{jk}^{uv} w_{ki}^{vu}\right) x_j, \\ &= (1-\lambda_u)x_i^0 + \lambda_u(1-\lambda_v)\sum_{k\in V} w_{ki}^{vu} y_k^0 + \lambda_u\lambda_v \sum_{j\in U} w_{ji}^{uu} x_j. \end{aligned} \tag{4.5}$$

The final scores of every entities can be obtained through an iteratively updating process. From our empirical testing, we find in most cases the equation can converge after about 10 iterations.

The proposed Co-HITS framework is general, and it contains a large algorithm space as shown in Table 4.1, in which HITS and personalized PageRank are actually two special cases in this space. If λ_u is set to be 0, the algorithm returns the initial scores as the *baseline*. If λ_u and λ_v are all equal to 1, Eq. (4.5) becomes the ordinary HITS equation,

$$x_i = \sum_{j \in U} w_{ji}^{uu} x_j. \tag{4.6}$$

If one of the parameters λ_u and λ_v is set to be 1, it can be regarded as the personalized PageRank (PPR) algorithm [54]. Suppose $\lambda_v = 1$, it becomes

$$x_i = (1 - \lambda_u) \cdot x_i^0 + \lambda_u \sum_{j \in U} w_{ji}^{uu} \cdot x_j. \tag{4.7}$$

When λ_v is set to be 0, the algorithm becomes a general hybrid method which aggregates the initial scores X^0 and Y^0 as follows,

$$x_i = (1 - \lambda_u) \cdot x_i^0 + \lambda_u \sum_{k \in V} w_{ki}^{vu} \cdot y_k^0, \tag{4.8}$$

which can be viewed as an one-step propagation algorithm.

4.3.2 Regularization Framework

Here we investigate a joint regularization framework for the above iterative framework. Let us first consider the vertices in one side, and imagine the personalized PageRank algorithm within the graph U as Eq. (4.7). For each iteration, every node receives the score from its neighbors (second term), and also retain its initial score (first term). The iteration process continues, and finally converges with the scores that are determined by their neighbors on the graph and their initial scores. A regularization framework can be developed

for the personalized PageRank algorithm, by regularizing the smoothness of relevance scores over the graph along with a regularizer on the initial ranking scores. The cost function R_1, associated with U, is defined to be

$$R_1 = \frac{1}{2}\sum_{i,j\in U} w_{ij}^{uu}\left\|\frac{x_i}{\sqrt{d_{ii}}} - \frac{x_j}{\sqrt{d_{jj}}}\right\|^2 + \mu\sum_{i\in U}\left\|x_i - x_i^0\right\|^2, \tag{4.9}$$

where $\mu > 0$ is the regularization parameter, and D is a diagonal matrix with entries $d_{ii} = \sum_j w_{ij}$ for normalization. Intuitively, the first term of the cost function defines the global consistency of the refined ranking scores over the graph, while the second term defines the constraint to fit the initial ranking scores, and the trade-off between each other can be controlled by the parameter μ. When $\mu \to +\infty$, R_1 puts all weights on the second term, and the regularization framework boils down to the *baseline* which corresponds to $\lambda_\mu = 0$ in Eq. (4.7). If $\mu = 0$, the regularization framework discards the initial ranking scores, and only takes into account the global consistency on the graph, which corresponds to $\lambda_\mu = 1$ in Eq. (4.7) (i.e., HITS as Eq. (4.6)). Similarly, for the cost function R_2 associated with V, we can show that

$$R_2 = \frac{1}{2}\sum_{i,j\in V} w_{ij}^{vv}\left\|\frac{y_i}{\sqrt{d_{ii}}} - \frac{y_j}{\sqrt{d_{jj}}}\right\|^2 + \mu\sum_{i\in V}\left\|y_i - y_i^0\right\|^2.$$

The intuition behind this framework is the global consistency, i.e., similar entities are most likely to have similar relevance scores with respect to a query.

Until now, R_1 and R_2 have defined the consistency based on the hidden links within U and V individually. However, the direct links between U and V may have more significant effect on the score propagation and mutual reinforcement. In this chapter, we investigate and develop a new cost function R_3 to consider the direct relationship between U and V:

$$R_3 = \frac{1}{2}\sum_{i\in U, j\in V} w_{ij}^{uv}\left\|\frac{x_i}{\sqrt{d_{ii}}} - \frac{y_j}{\sqrt{d_{jj}}}\right\|^2 + \frac{1}{2}\sum_{j\in V, i\in U} w_{ji}^{vu}\left\|\frac{y_j}{\sqrt{d_{jj}}} - \frac{x_i}{\sqrt{d_{ii}}}\right\|^2 \tag{4.10}$$

The intuition behind R_3 is the smoothness constraint between two entity sets, which penalizes large differences in relevance scores for vertices between U and V that are strongly connected.

Formally, the cost function R, associated with both U and V, is defined to be

$$R = \lambda_r(R_1 + \alpha R_2) + (1 - \lambda_r)R_3, \tag{4.11}$$

where $\alpha > 0$ and $\lambda_r \in [0, 1]$. By minimizing the cost function R, we obtain the general regularization framework associated with the general Co-HITS equation as Eq. (4.5). In this chapter, we simply set $\alpha = 1$ and focus on investigating the effect of parameter λ_r. Then the original optimization problem $\min_F(R)$ can be rewritten as follows:

$$\begin{aligned} \min_{F} \quad & \frac{1}{2}\sum_{i,j=1}^{m+n} w_{ij} \left\| \frac{f_i}{\sqrt{d_{ii}}} - \frac{f_j}{\sqrt{d_{jj}}} \right\|^2 + \mu \sum_{i=1}^{m+n} \left\| f_i - f_i^0 \right\|^2 \\ s.t. \quad & W = \begin{bmatrix} W^{uu} & \beta \cdot W^{uv} \\ \beta \cdot W^{vu} & W^{vv} \end{bmatrix} \\ & F = \begin{bmatrix} X \\ Y \end{bmatrix} \\ & \beta = (1 - \lambda_r)/\lambda_r, \end{aligned} \tag{4.12}$$

where X and Y are the score vectors for U and V respectively. Differentiating Eq. (4.12) [151, 156], we have

$$\left.\frac{dR}{dF}\right|_{F=F^*} = F^* - SF^* + \mu(F^* - F^0) = 0, \tag{4.13}$$

where $S = D^{-\frac{1}{2}}WD^{-\frac{1}{2}}$, then Eq. (4.13) can be transformed into

$$F^* - \frac{1}{1+\mu}SF^* - \frac{\mu}{1+\mu}F^0 = 0. \tag{4.14}$$

Table 4.1: Connections with other methods

Iterative Framework		
λ_u	λ_v	Description
$= 0$	$\in [0, 1]$	Initial scores $x_i = x_i^0$
$= 1$	$= 1$	Original HITS as Eq. (4.6)
$\in (0, 1)$	$= 1$	Personalized PageRank as Eq. (4.7)
$\in (0, 1)$	$= 0$	One-step propagation as Eq. (4.8)
$\in (0, 1)$	$\in (0, 1)$	General Co-HITS as Eq. (4.5)

Regularization Framework	
μ_α, λ_r	Description
$\mu_\alpha = 0$	Initial scores $x_i = x_i^0$
$\mu_\alpha = 1$	Corresponding to HITS
$\mu_\alpha \in (0, 1)$	General regularization framework
$\lambda_r = 1$	Single-sided regularization
$\lambda_r \in (0, 1)$	Double-sided regularization
$\lambda_r = 0.5$	$R = 0.5(R_1 + R_2) + 0.5R_3$

After simplifying, a closed-form solution can be derived,

$$F^* = \mu_\beta (I - \mu_\alpha S)^{-1} F^0, \tag{4.15}$$
$$\mu_\alpha = \frac{1}{1+\mu}, \text{ and } \mu_\beta = \frac{\mu}{1+\mu},$$

where I is an identity matrix. Note that μ_α ranges from 0 to 1, and $\mu_\alpha + \mu_\beta = 1$. In this chapter, we consider the normalized Laplacian in [151], and S is positive-semidefinite. Details about how to calculate the matrix W and S will be introduced in Section 4.4.1. Given the initial ranking scores F^0 and the matrix S, we can compute the refined ranking scores F^* directly.

4.3.3 Connections and Justifications

In this section, we establish connections between the generalized Co-HITS algorithm and other methods in Table 4.1. The iterative framework contains HITS, personalized PageRank, and the one-step propagation algorithm as the special cases. When looking at the regularization framework, its variations are controlled by the parameters μ_α and λ_r. When $\mu_\alpha = 0$ ($\mu \to +\infty$), R puts all weights on the second term, and the regularization framework boils down to the *baseline*. If $\mu_\alpha = 1$ ($\mu = 0$), the regularization framework discards the initial ranking scores, and only takes into account the global consistency on the graph, which corresponds to the HITS algorithm. Moreover, a different selection of λ_r leads to a different smoothing strategy. If $\lambda_r = 1$, it only considers the single-side regularization within U and V. If $\lambda_r \in (0, 1)$, it utilizes the double-side regularization to make full use of the bipartite graph.

For the large-scale information retrieval, the matrix S is usually very large but sparse, which can be loaded in a relatively small storage space. However, the inverse matrix $(I - \mu_\alpha S)^{-1}$ will be very dense, and may need a huge space to save it. To balance the storage space and the computation time of the inverse matrix, we suggest to approximate the Eq. (4.15) in a specific subgraph with a submatrix $\hat{S}$, which consists of the top-n entities according to the initial ranking scores $\hat{F}^0$. It can be found that the top ranking scores usually outnumber the very low ranking scores. Theoretically, if the ranking scores after n are close to 0, the following approximate solution is equivalent to Eq. (4.15),

$$\hat{F}^* = (I - \mu_\alpha \hat{S})^{-1} \hat{F}^0. \tag{4.16}$$

In this equation, we eliminate the parameter μ_β as it does not change the ranking. Accordingly, it needs to calculate the inverse matrix $(I - \mu_\alpha \hat{S})^{-1}$ online. Fortunately, the matrix is usually very sparse, then the complexity time of the sparse matrix inversion can be reduced to be linear with the

number of nonzero matrix elements. In our experiments, we extract the top 5,000 entities for approximation.

4.4 Application to Bipartite Graphs

To illustrate our proposed method, we use the statistical language model as the baseline to calculate the initial relevance scores based on the content information, and specify the application in query suggestion base on the query-URL bipartite graph. In this section we introduce the bipartite graph construction and the statistical language model, then show the overall algorithm of our framework.

4.4.1 Bipartite Graph Construction

Bipartite graphs are widely used to describe the relationship between queries U and URLs V when mining the query logs, such as query suggestion and classification. The edges of the query-URL bipartite graph can capture some semantic relations between queries and URLs. For each edge $(q_i, d_j) \in E$ we associate a numeric weight c_{ij}, known as the *click frequency*, that measures the number of times the URL d_j was clicked when shown in response to the query q_i. The transition probability w_{ij}^{uv} [31, 107] from the query q_i to the URL d_j is defined by normalizing the click frequency from the query q_i as

$$w_{ij}^{uv} = \frac{c_{ij}}{\sum_{j \in V} c_{ij}},$$

while the transition probability w_{ji}^{vu} from the URL d_j to the query q_i is defined as

$$w_{ji}^{vu} = \frac{c_{ij}}{\sum_{i \in U} c_{ij}}.$$

Thus, we can easily obtain the transition matrices W^{uv}, W^{vu}, W^{uu} and W^{vv}.

In practice, it is sometimes unnecessary to apply our learning algorithms to a very large bipartite graph constructed from the entire collection. Since

our task is to find the most relevant queries as suggestion for a given query, it would be more efficient to apply our algorithm only to a relatively compact query-URL bipartite graph that covers the relevant queries and related URLs. We utilize the same method used in [81] for building a compact query-URL bipartite graph and iteratively expanding it in the following,

1. Initialize a query set $\hat{U} = U_L$ (seed query set), and initialize a URL set $\hat{V} = V_L$ (seed URL set);
2. Update $\hat{V}$ to add the set of URLs that are connected with $\hat{U}$;
3. Update $\hat{U}$ to be the set of queries that are connected with $\hat{V}$;
4. Iterate 2 and 3 until $\hat{U}$ and $\hat{V}$ reach a desired size;

The final bipartite graph $\hat{G}$ to which the algorithms are applied consists of $\hat{U}$, $\hat{V}$ and edges $\hat{E}$ connecting them. According to the relevance scores, we initialize the top-10 relevant queries and top-10 relevant URLs as the seed sets. Generally, it only needs one iteration to reach 5,000 entities in our experiments. In this chapter, we employ the widely used k-nearest neighbor (k-NN) graph, where each node is connected to its k nearest neighbors under the transition probability measure and the edges can be weighed by the transition matrices. It has been shown to be effective when $k = 10$ in [40]. Then, the matrix $\hat{W}$ is constructed with maximum 50,000 $(5,000 \times 10)$ entries. After normalization, we can obtain the matrix $\hat{S}$. Fortunately, the matrix is usually very sparse, and the complexity time of the sparse matrix inversion can be reduced to be linear with the number of nonzero matrix elements.

4.4.2 Statistical Language Model

Using language models for information retrieval has been studied extensively in recent years [108, 147, 148]. To determine the probability of a query given a document, we infer a document model θ_d for each document in a

collection. With query q as input, retrieved documents are ranked based on the probability that the document's language model would generate the terms of the query, $p(q|\theta_d)$. The ranking function $f^0(q,d)$ can be written as

$$f^0(q,d) = p(q|\theta_d) = \prod_{t \in q} p(t|\theta_d)^{n(t,q)}, \tag{4.17}$$

where $p(t|\theta_d)$ is the maximum likelihood estimation of the term t in a document d, and $n(t,q)$ is the number of times that term t occurs in query q. The likelihood of a query q consisting of a number of terms t for a document d under a language model with Jelinek-Mercer smoothing [148] is $p(t|\theta_d) = 0.5p(t|d) + 0.5p(t)$. With the language model, we calculate the initial ranking scores of the documents with respect to a query.

In our proposed method, we employ the language model to determine the initial relevance scores F^0 for the queries and URLs. Note the queries from the query log are very short, but it still can be viewed as a document in the language model. We can get better initial relevance scores if we perform the query expansion and construct the document model with the expanded queries. For each URL, although its exact content information is not included in the query log, it can be represented as a document by the aggregation of connected queries [106].

4.4.3 Overall Algorithm

By unifying the Co-HITS algorithm in Section 4.3 and the application to bipartite graphs, we summarize the proposed algorithm in Algorithm 1. In the algorithm, note that we first perform preprocessing in a collection to construct the bipartite graph, and calculate the transition matrices. In the algorithm, we calculate the initial ranking scores using the language model, extract the compact bipartite subgraph, and perform the Co-HITS algorithm.

To implement the Co-HITS algorithm, we employ a sparse matrix package, i.e., CSparse [34], to solve the sparse matrix inversion efficiently. To deploy

Algorithm 1 Generalized Co-HITS Algorithm

Input: Given a query q and the bipartite graph

Perform:

1. Calculate the initial ranking scores based on the statistical language model and extract the top-ranked U_L and V_L as the seed sets;
2. Expand and extract the compact bipartite subgraph $\hat{G} = (\hat{U} \cup \hat{V}, \hat{E})$;
3. Get the weight matrix $\hat{W}$ or $\hat{S}$, and normalize the corresponding initial scores F^0;
4. Solve Eq. (4.5) or Eq. (4.16) and get the final scores $\hat{F}^*$.

Output: Return the ranked queries.

the efficient implementations of our scheme, all of the other algorithms used in the study are programmed in the C# language. We have implemented the language modeling approach to obtain the initial relevance scores with the Lucene.Net[1] package. For these experiments, the system indexes the collection and does tokenization, stopping and stemming in the usual way. The testing hardware environment is on a Windows workstation with 3.0GHz CPU and 1GB physical memory.

4.5 Experimental Evaluation

In the following experiments we compare our proposed algorithm with other methods on the tasks of mining query logs through an empirical evaluation. We define the following task: Given a query and a query-URL bipartite graph, the system has to identify a list of queries which are most similar or semantically relevant to the given query. In the rest of this section, we introduce the data collection, the assessments and evaluation metrics, and present the

[1] http://incubator.apache.org/lucene.net/

Table 4.2: Samples of the AOL query log dataset.

UserID	Query	Time	Rank	ClickURL
2722	yahoo	2006-04-25 13:03:23	1	http://www.yahoo.com
121537	map	2006-05-25 18:28:58	1	http://www.mapquest.com
123557	travel	2006-03-13 01:09:53	2	http://www.expedia.com
1903540	cheap flight	2006-05-15 00:31:43	1	http://www.cheapflights.com

experimental results.

4.5.1 Data Collection

The dataset that we study is adopted from the query log of AOL search engine [102]. The entire collection consists of $19,442,629$ user click-through records. These records contain $10,154,742$ unique queries and $1,632,789$ unique URLs submitted from about $650,000$ users over three months (from March to May 2006). As shown in Table 4.2, each record of the click contains the same information: UserID, Query, Time, Rank and ClickURL. This dataset is the raw data recorded by the search engine, and contains a lot of noises. Hence, we conduct a similar method employed in [137] to clean the raw data. We clean the data by removing the queries that appear less than 2 times, and by combining the near-duplicated queries which have the same terms without the stopwords and punctuation marks (for example, "google's image" and "google image" will be combined as the same query). After cleaning, our data collection consists of $883,913$ queries and $967,174$ URLs. After the construction of the click graph, we observe that a total of $4,900,387$ edges exist, which indicates that each query has 5.54 distinct clicks, and each URL is clicked by 5.07 distinct queries. Moreover, taken as a whole, this data collection has $250,127$ unique terms which appear in all the queries.

4.5.2 Assessments and Evaluation Metrics

It is difficult to evaluate the quality of query similarity/relevance rankings due to the scarcity of data that can be examined publicly. For an automatic evaluation, we utilize the same method used in [7] to evaluate the similarity of retrieved queries, but engage the Google Directory[2] instead of the Open Directory Project[3]. When a user types a query in Google Directory, besides site matches, we can also find *category* matches in the form of paths between directories. Moreover, these categories are ordered by relevance. For instance, the query "United States" would provide the hierarchical category "Regional > North America > United States", while one of the results for "National Parks" would be "Regional > North America > United States > Travel and Tourism > National Parks and Monuments". Hence, to measure how similar two queries are, we can use a notion of similarity between the corresponding categories provided by the search results of Google Directory. In particular, we measure the similarity between two categories Ca_i and Ca_r as the length of their longest common prefix $P(Ca_i, Ca_r)$ divided by the length of the longest path between Ca_i and Ca_r. More precisely, the similarity is defined as:

$$Sim(Ca_i, Ca_r) = \frac{|P(Ca_i, Ca_r)|}{\max(|Ca_i|, |Ca_r|)}, \tag{4.18}$$

where $|Ca_i|$ denotes the length of a path. For instance, the similarity between the above two queries is 3/5 since they share the path "Regional > North America > United States" and the longest one is made of five directories. We evaluate the similarity between two queries by measuring the similarity between the aggregated categories of the two queries, among the top 5 answers provided by Google Directory.

To give a fair assessment, we randomly select 300 distinct queries from the data collection, then retrieve a list of similar queries using the proposed

[2]http://directory.google.com/

[3]http://www.dmoz.org/

methods for each of these queries. For the evaluation of the task, we adopt the precision at rank n to measure the relevance of the top n results of the retrieved list with respect to a given query q_r, which is defined as

$$P@n = \frac{\sum_{i=1}^{n} Sim(q_i, q_r)}{n}, \tag{4.19}$$

where $Sim(q_i, q_r)$ means the similarity between q_i and q_r. In our experiments, we report the precision from $P@1$ to $P@10$, and take the average over all the 300 distinct queries.

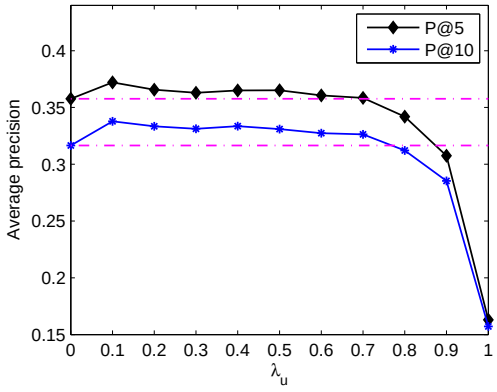
4.5.3 Experimental Results

We consider the question whether our proposed method can boost the performance using the generalized Co-HITS algorithm for query suggestion. First the experiments are performed to compare the iterative framework of Co-HITS with different parameters λ_u and λ_v. Then we examine the performance of the regularization framework by varying the parameters μ_α and λ_r. Finally, we investigate and compare the detailed results of different methods, which shows that the regularization framework CoRegu-0.5 achieves the best results.

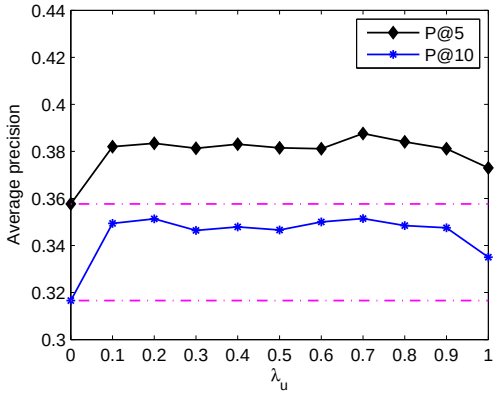
Comparison of Iterative Framework

For the iterative framework, the generalized Co-HITS contains HITS, personalized PageRank (PPR), and the one-step propagation (OSP) algorithms as the special cases. In this subsection, we compare the performance of general Co-HITS (CoIter) with the above special cases, and report the precisions of P@5 and P@10 in Figure 4.3.

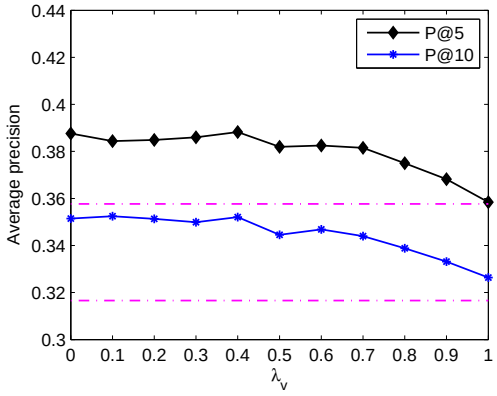
First of all, we evaluate the performance of personalized PageRank after setting $\lambda_v = 1$. Figure 4.3(a) illustrates the experimental results for different λ_u, in which the solid curves indicate the precisions of P@5 and P@10 for



(a) $\lambda_v = 1$ (PPR)



(b) $\lambda_v = 0$ (OSP)



(c) $\lambda_u = 0.7$ (CoIter)

Figure 4.3: The effect of varying parameters (λ_u and λ_v) in the iteration framework: (a) personalized PageRank, (b) one-step propagation, and (c) general Co-HITS. The dashed lines denote the baseline results.

different parameters, and the dashed curves denote the precisions for the *baseline*. We can see that the performance has only a slight increase when compared to the baseline if λ_u is set close to 0. With the increase of λ_u, the performance becomes worse, and even underperforms the baseline. It is because of the lack of relevance constraints from both sides of the bipartite graph, so the score propagation on the graph may be influenced easily due to some noise edges. When λ_u is equal to 1, it corresponds to the HITS algorithm that discards the initial relevance scores.

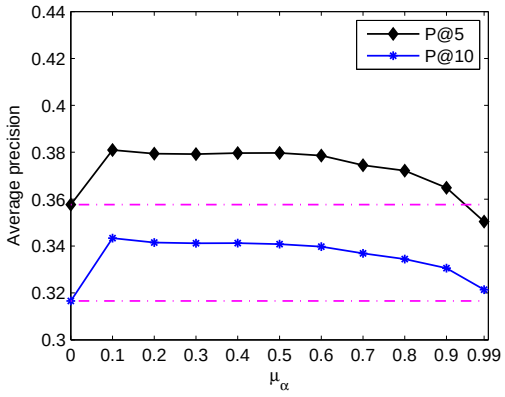
When $\lambda_v = 0$, the Co-HITS algorithm boils down to simply aggregation of the initial scores from both sides. As shown in Figure 4.3(b), we notice that the simple aggregation method (i.e., one-step propagation when λ_u is set from 0.1 to 0.9) benefits from both sides, and outperforms the method that only considers from one side. This observation supports the intuition of our Co-HITS algorithm that the initial relevance scores from both sides provide valuable and reinforced information as well as the constraints of relevance.

To illustrate the performance of general Co-HITS algorithm, we choose to set $\lambda_u = 0.7$ and vary the parameter λ_v from 0 to 1, and then show the results in Figure 4.3(c). From this figure, we can observe that its improvement over the baseline is promising when compared to the personalized PageRank, and it is comparable with the one-step propagation when λ_v is set to be 0.4.

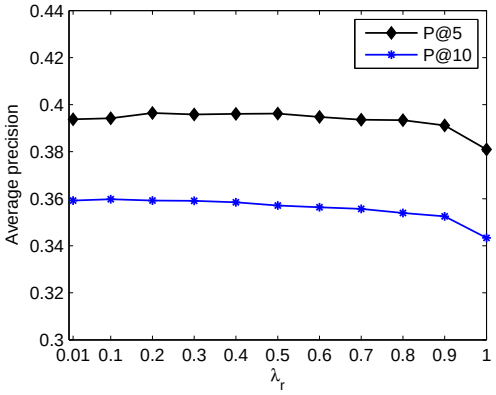
Comparison of Regularization Framework

For the regularization framework, we first evaluate the single-sided regularization (SiRegu) by varying the parameter μ_α, then we fix μ_α and perform the double-sided regularization (CoRegu) with different λ_r.

As mentioned in Table 4.1, the parameter μ_α is used to control the balance between the global consistency and the initial ranking scores in the unified regularization framework as Eq. (4.9), and it ranges from 0 to 1. The



(a) $\lambda_r = 1$ (SiRegu)



(b) $\mu_\alpha = 0.1$ (CoRegu)

Figure 4.4: The effect of varying parameters (μ_α and λ_r) in the regularization framework: (a) single-sided regularization, and (b) double-sided regularization.

experimental results for the single-sided regularization are illustrated in Figure 4.4(a). When $\mu_\alpha = 0$, SiRegu boils down to the initial *baseline*. We can see that the performance is improved over the *baseline* when incorporating the global consistency ($\mu_\alpha > 0$) in the framework. With the increase of μ_α, the performance becomes better until it puts too much weight on the term of global consistency ($\mu_\alpha \to 1$). If $\mu_\alpha \to 1$, SiRegu discards the initial ranking scores, and only takes into account the global consistency on the graph. As shown in Figure 4.4(a), when the parameter μ_α is equal to 0.99, the performance of our method becomes worse than the initial *baseline* due to the overweighted global consistency. According to the theoretical analysis in Section 4.3.2, SiRegu corresponds to the personalized PageRank in the iteration framework. By comparing Figure 4.4(a) with Figure 4.3(a), both results are improved first and then degraded with the increase of μ_α and λ_u, which shows that the parameters μ_α and λ_u have similar impact on SiRegu and PPR, respectively.

We have shown that SiRegu can improve the performance over the initial baseline, and achieves the best performance when μ_α is set to be 0.1. Now we fix $\mu_\alpha = 0.1$, and examine whether CoRegu can further boost the performance by incorporating a direct smoothness constraint between two entity sets. According to Figure 4.4(b), it is obvious that CoRegu ($\lambda_r < 1$) performs better than SiRegu ($\lambda_r = 1$). The improvement over the SiRegu method owes to the direct smoothness constraint as Eq. (4.10) which is incorporated in the CoRegu framework. This observation supports the theoretical analysis of the proposed regularization framework. Moreover, CoRegu is relatively robust and may achieve the best results when the parameter λ_r is set to be 0.2-0.6.

Table 4.3: Comparison of different methods by P@5 and P@10. The mean precisions and the percentages of relative improvements are shown in the table.

Method	Para		Evaluation metrics	
Iter	λ_u	λ_v	P@5	P@10
Baseline	0	$\times$	0.358 (0%)	0.317 (0%)
PPR-0.1	0.1	1	0.372 (4.0%)	0.338 (6.7%)
OSP-0.7	0.7	0	0.388 (8.4%)	0.351 (11.0%)
CoIter-0.4	0.7	0.4	0.388 (8.6%)	0.352 (11.2%)
Regu	λ_r	μ_α	P@5	P@10
SiRegu-0.1	1	0.1	0.381 (6.5%)	0.343 (8.5%)
CoRegu-0.5	0.5	0.1	0.396 (10.8%)	0.357 (12.8%)

Detailed Results

To gain a better insight into the proposed Co-HITS algorithm, we compare the best results of different models using P@5 and P@10 in Table 4.3. The mean precisions and the percentages of relative improvements over the baseline are shown in the table. A quick scan of the table reveals that CoRegu-0.5 achieves the best performance. When looking at the relative improvements of those models, we can see that CoRegu-0.5 improves over the baseline by 10.8% (for P@5) and 12.8% (for P@10) respectively, while CoIter-0.4 over the baseline by 8.6% and 11.2%. In addition, SiRegu-0.1 performs better than PPR-0.1. These results confirm that the regularization framework outperforms the iterative framework.

Figure 4.5 illustrates the precisions of six models from P@1 to P@10. In general, we can see that the performances of all the models, except the

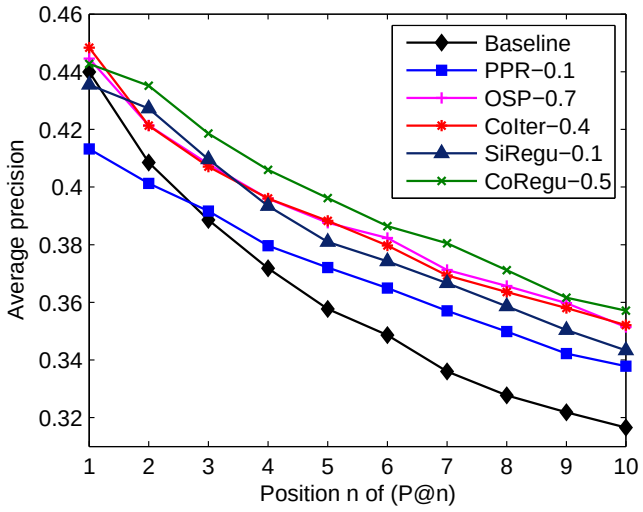


Figure 4.5: Comparison of six models.

PPR-0.1, are better than the baseline. It is comparable for the precisions of OSP-0.7, CoIter-0.4 and SiRegu-0.1. The double-sided regularization model, i.e., CoRegu-0.5, achieves the best performance, whose improvements are consistent. After looking into the details, one important observation is that the improvements of our method over the baseline are increased for larger n (of the evaluation matric P@n). This is because the mutual reinforcement can boost the semantically relevant entities which have low initial scores. According to all the the experimental results, we can argue that it is very essential and promising to consider the double-sided regularization framework for the bipartite graph.

4.6 Related Work and Discussions

The work is related to the category of link analysis methods. In [43], the authors have tried to model a unified framework for link analysis, which includes the two popular ranking algorithms HITS [69] and PageRank [18]. Several normalized ranking algorithms are studied which are intermediate between

HITS and PageRank. Our method differs from this unified framework as we integrate the graph information with the content information.

According to some generalization of PageRank and HITS, a family of work on the structural re-ranking paradigm over a graph was proposed to refine the initial ranking scores. Kurland and Lee performed re-ranking based on centrality within graphs, through PageRank-inspired algorithm [72] and HITS-style cluster-based approach [73]. Zhang et al. [149] proposed a similar method to improve Web search results based on a linear combination of results from text search and authority ranking. In addition, PopRank [101] is developed to extend PageRank models to integrate heterogenous relationships between objects. Another approach suggested by Minkov et al. [98] has been used to improve an initial ranking on graph walks in entity-relation networks. However, those methods does not make full use of the content and the graph information as they treat the content and the graph information individually.

The regularization framework we proposed is closely related to graph-based semi-supervised learning [132, 151, 153, 156], which usually assume label smoothness over the graph. Mei et al. [95] extend the graph harmonic function [156] to multiple classes. However, our work is different from theirs, as their tasks are mainly used in query-independent settings (i.e., semi-supervised classification, topic modeling), while we focus on query-dependent ranking problems. With the advance of machine learning, graph-based models have been widely and successively used in information retrieval and data mining. Diaz [42] use score regularization to adjust ad-hoc retrieval scores from an initial retrieval. Deng et al. [40] propose a method to learn a latent space graph from multiple relationships between objects, and then regularize the smoothness of ranking scores over the latent graph. More recently, Qin et al. [109] use relational objects to enhance learning to rank with parameterized

regularization models. But those three methods only consider the regularization from one side of the bipartite graph or within a single graph, while our regularization framework takes into account not only the smoothness within the same entity set but also the direct relationship between two entity sets.

This work is also related to query log analysis [7], as we apply our Co-HITS algorithm to the application of query suggestion by mining the query logs. A common model for utilizing query logs from search engines is in the form of a query-URL bipartite graph (i.e., click graph) [31]. Based on the click graph, many research efforts in query log analysis have been devoted to query clustering [11], query suggestion [66, 86] and query classification [81]. Craswell and Szummer [31] used click graph random walks for relevance rank in image search. Mei et al. [96] proposed an approach to query suggestion by computing the hitting time on a click graph. Li et al. [81] presented the use of click graphs in improving query intent classifiers. In this work, we combine the click graph with the content information from queries and URLs to improve the precisions of the results, which differs from the previous methods.

4.7 Summary

In this chapter we have presented the generalized Co-HITS algorithm for bipartite graphs, whose basic idea is to incorporate the bipartite graph with the content information from both sides. We not only formally define the iterative framework, but also investigate the regularization framework for the generalized Co-HITS algorithm from different views. For the iterative framework, it has been shown that HITS, personalized PageRank, and the one-step propagation algorithm are special cases of the generalized Co-HITS algorithm. In the regularization framework, we successfully build the connection with HITS, and develop a new cost function to consider the direct relationship between two entity sets, which leads to a significant improvement over the

baseline method. We have applied the proposed algorithm to mine the query log and compare with many different settings. Experimental results show that the improvements of our proposed model are consistent, and CoRegu-0.5 achieves the best performance. In future work, it would be interesting to investigate the performance of our Co-HITS algorithm in other bipartite graphs to see if the proposed method might have an impact on any bipartite graphs.

□ **End of chapter.**

Chapter 5

Modeling Expertise Retrieval

The objective of this thesis is to propose a general Web mining framework to combine the content with the graph information as well as other kinds of information effectively. In previous two chapters, we have described several models and their applications to query log analysis by combining the content and graph information. In the following two chapters, we will address the high-level expertise retrieval task (which is also called expert finding) and investigate several models to incorporate different information in a more heterogeneous information environment.

In this chapter, we aim to address expert finding task in a real-world academic field. We propose a novel expert finding framework based on the large-scale DBLP bibliography and its supplemental data from Google Scholar. We formally define a weighted language model to aggregate the expertise of a candidate from the associated documents. The model not only considers the relevance of documents against a given query, but also incorporates the importance of documents in the form of document priors. Moreover, we investigate and integrate a graph-based regularization method, which can be viewed as a special case of the Co-HITS algorithm, to boost our model by refining the relevance scores of the documents with respect to the query.

5.1 Problem and Motivation

With the development of information retrieval techniques, many research efforts in this field have been made to address high-level information retrieval and not just the traditional document retrieval, such as entity retrieval [104, 144] and expertise retrieval [9]. Since the advent of the expert finding task in the TREC Enterprise track [30, 133], expertise retrieval (i.e., expert finding) has received increased interests in both industry and academia. The task of expert finding is to retrieve a ranked list of persons that possess expertise on a given topic. Most current developments in expert search are concentrated in the Enterprise corpora, as TREC2005 [30] and TREC2006 [133] have provided a common platform for researchers to empirically assess methods and techniques devised for expert finding. However, there is a lack of research work for expert search in a specific academic field. Identification of the experts for a particular academic topic could be of great value. There are many important research topics and practical applications, for example, recommending panels of reviewers for state research grant applications [56], determining important experts for consultation by researchers embarking on a new research field [93], and assigning papers to reviewers automatically in a peer-review process [116, 97].

Before we introduce the approach to search experts automatically, let us imagine the way that researchers identify experts for a specific research topic. One natural way is to first retrieve articles related to the topic, then examine the authors of those articles and determine the experts with human judgments. Figure 5.1 presents an example to artificially identify experts that have expertise on the topic "probabilistic relevance model." Using the topic as the query to search in Google Scholar [2], it will be easy to obtain the relevant articles related to the query topic. Based on retrieved records, the researchers "Stephen E. Robertson" and "C. J. van Rijsbergen" may be

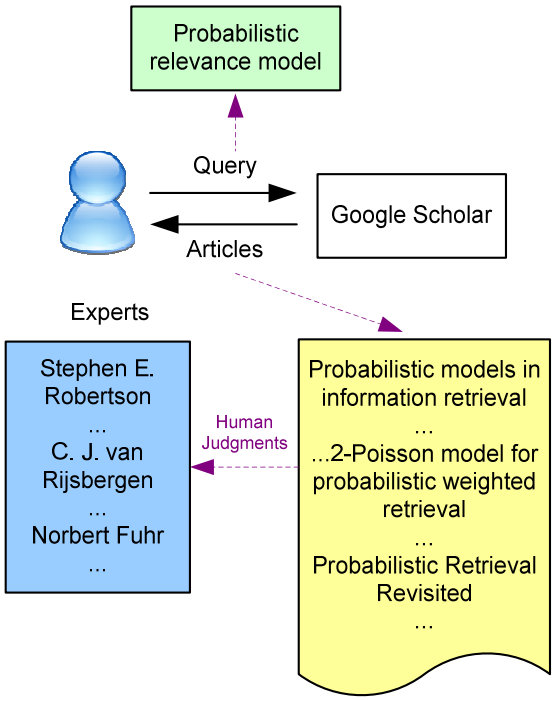


Figure 5.1: A sample of the artificial expert search process.

identified as the experts in this case. However, it is not easy for an outsider or a novice to identify the most important experts in a new research field. It usually requires the prior knowledge of this topic for the user to make the right judgment about the experts. The intensive manual labor search prompted us to design a system to search for experts automatically.

Expert finding has been treated as an information retrieval task in previous approaches [8, 21, 30, 133]. One of the state-of-the-art approaches [8, 9] is based on document-based model using a statistical language model to rank experts (we refer to it as our *baseline model*). In this chapter, we propose a novel framework to aggregate the expertise of a candidate based on the relevance and importance of the associated documents. More specifically, we formally define a weighted language model that takes into consideration not only the relevance between a query and documents but also the importance

of the documents. Suppose there are two documents d_1 and d_2 with the same relevance scores to a query q, while they have different importance and d_1 is more important than d_2. Intuitively, it is more reasonable that ca_1 (the author of d_1) has the higher probability of being an expert than ca_2 (the author of d_2) with respect to the query. The underlying idea of our model is *that the more important the document is, the higher prior probability it is written by an expert.* In contrast to existing methods [8, 48, 103], this probability is ignored or assumed to be uniform. However, in our approach, such a prior probability is estimated based on the citation number for each document, and it is simultaneously integrated with the relevance scores.

One of the key issues for finding experts is to retrieve the most relevant documents along with the relevance scores. However, the initial relevance scores, estimated by the basic language model, tend to be imperfect. In order to estimate the relevance scores more correctly, a graph-based approach is proposed to refine the relevance scores by regularizing the smoothness of the relevance scores on the graph along with a regularizer on the initial relevance scores. The graph is either constructed by explicit link structures [18, 69], such as hyperlinks of Web pages, citations of research papers, or inferred from the content information, such as k-nearest neighbor graph [72] and affinity graph [149]. The intuition behind the model is the global consistency on the graph: *Similar documents are likely to exhibit the same relevance scores with respect to a query.* In other words, if the neighbors of a document are highly relevant to a query, this document is most likely to be relevant to the query; otherwise, if none of the neighbors of a document is relevant to the query, the document is unlikely to be relevant to the query. With such a graph-based regularization method, the performance of our weighted language model may be further improved.

The schematic of our expert finding system is illustrated in Figure 5.2.

Our approach consists of the following three major steps. First, when a query is submitted to the system, top-n related papers are retrieved by the literature retrieval component, in which the initial relevance and importance scores are obtained for the relevant articles. Second, a graph-based regularization method is employed to refine the initial relevance scores. Finally, the expertise of a candidate is aggregated based on the relevance and importance of associated documents, then the ranked experts with respect to the query are returned to the user.

To deal with the expert-finding task in a real-world academic field, an essential component is therefore the acquisition of a dataset replete with publications from which expertise can be accessed. The DBLP bibliography [1] is a good starting point for extracting the data needed for this application, as it contains more than 955,000 articles with over 574,000 authors from conferences and journals in the Computer Science field. We could construct a paper-author bipartite graph based on the DBLP bibliography data, which directly builds the document-candidate associations. In scientific research, we make the assumption that the expertise of a researcher could be represented by his/her publications [116]. One limitation of DBLP bibliography data is that each record only contains the paper title without the abstract and index terms. The information provided by the title is too limited to represent the paper and to calculate the relevance scores between papers and queries. To address this problem, Google Scholar [2] is utilized as data supplementation, thus each paper record is represented by the combination of the paper title and its supplemental data.

To illustrate our methodology, we compare our weighted language model with the baseline model, and investigate the effectiveness of the graph-based regularization method. According to the experimental results, our weighted language model performs much better than the baseline model, which indi-

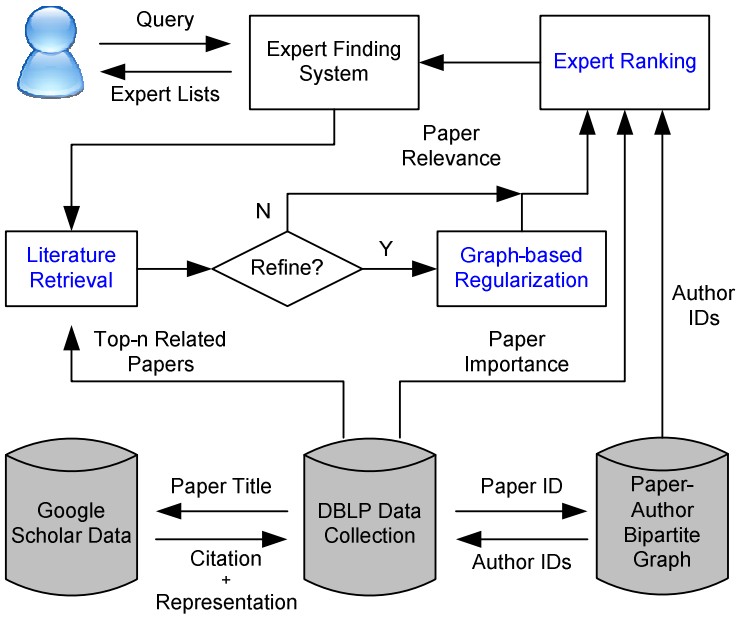


Figure 5.2: The schematic of general expert finding systems.

cates that it is very important to consider the prior probability in the model. Moreover, experimental results demonstrate the weighted language model can be further enhanced with the graph-based regularization method, and the improvements are consistent and promising.

The rest of this chapter is organized as follows. Section 5.2 provides detailed descriptions of the expertise modeling based on the language model, as well as the overall algorithm. Section 5.3 presents the graph-based regularization model which is used to refine the initial relevance scores. Section 5.4 defines the experimental setup of our methods. Section 5.5 evaluates the experimental results. Finally, Section 5.6 summarizes this chapter.

5.2 Modeling Expert Search

In this section, we detail the expert finding task in the academic domain, and propose a generative probabilistic model to identify the expert researchers.

Generally, influential researchers publish many manuscripts in their fields. Therefore, their expertise could thus be deduced based on the overall aggregation of their publications. For a given query, the basic idea is to model an expert candidate based on the relevance and importance of associated documents.

5.2.1 Problem Definition

Formally, suppose $CA = \{ca_1, ca_2, ..., ca_m\}$ is the set of expert candidates to be retrieved. Let $D = \{d_1, d_2, ..., d_n\}$ denote a collection of supporting documents, where d_i is a paper authored by one or several candidates. Given a query q, we formulate the problem of identifying experts using a generative probabilistic model, i.e., what is the probability of a candidate ca being an expert given the query topic q?

Specifically, the task is to determine $p(ca|q)$, and rank candidates ca according to this probability that a candidate is "relevant" to the topic (i.e., expertise) specified in a query. Using Bayes' theorem, the probability can be formulated as follows:

$$p(ca|q) = \frac{p(ca, q)}{p(q)}, \tag{5.1}$$

where $p(ca, q)$ is the joint probability of a candidate and a query, $p(q)$ is the probability of the query. The probability $p(q)$ is a constant, so it can be ignored for ranking purposes. To calculate the probability $p(ca|q)$, it is equivalent to estimate the joint probability $p(ca, q)$. The generative probabilistic model used to estimate the probability $p(ca, q)$ can be defined as follows:

$$\begin{aligned} p(ca, q) &= \sum_{d \in D} p(d)p(ca, q|d) \\ &= \sum_{d \in D} p(d)p(q|d)p(ca|d, q), \end{aligned} \tag{5.2}$$

where $p(d)$ is the prior probability of a document, $p(q|d)$ means the relevance between q and d, and $p(ca|d, q)$ represents the association between the can-

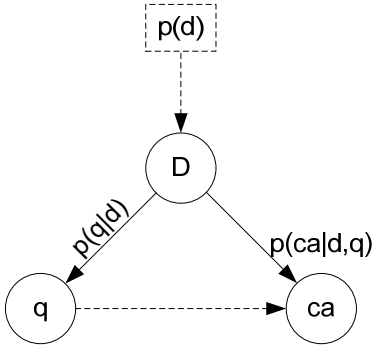


Figure 5.3: The weighted model for expert finding.

didates and the documents for a given query. As shown in Figure 5.3, the supporting documents D act as a "bridge" to connect the query q with the candidate ca.

Under this model, the process of finding an expert is as follows: Given a collection of documents ranked according to the query, we first examine each document relevant to the query, and then identify the authors associated with that document. Finally, the expertise of a candidate is deduced based on the overall aggregation of the relevance as well as the priors of the associated documents. In the process, it is shown that our model aims to search experts the way researchers do. Here, the automatic search process is taken to the extreme where we consider all documents in the collection.

In the following subsections, we will present the process with three main components and discuss how to estimate them respectively.

5.2.2 Paper Relevance

In order to retrieve the most relevant documents, the key challenge is to compute the relevance between a query and documents. In recent years, statistical language model has been widely used in the application of information retrieval [60, 108, 145, 148]. The basic idea of these approaches is to estimate

a language model for each document, and then rank documents by the likelihood of their matching the query according to the estimated language model.

To determine the probability of a query given a document, we infer a document language model θ_d for each document. The relevance score of document d with respect to query q is then defined as the conditional probability $p(q|\theta_d)$. Suppose $q = t_1...t_m$ and each term t is generated independently, the relevance score would be,

$$f(q,d) = p(q|\theta_d) = \prod_{t \in q} p(t|\theta_d)^{c(t,q)}, \tag{5.3}$$

where $c(t,q)$ is the count of term t in query q, and $p(t|\theta_d)$ is the maximum likelihood estimator of the term in a document d.

With such a model, the retrieval problem is reduced to the problem of estimating $p(t_i|\theta_d)$. In order to assign nonzero probabilities to unseen words, we adopt the Dirichlet prior smoothing method [148] to estimate the term likelihood with the collection language model:

$$p(t|\theta_d) = \frac{c(t,d) + \nu p(t|C)}{|d| + \nu}. \tag{5.4}$$

where ν is the parameter to control the amount of smoothing, and $p(t|C)$ is the collection language model.

Based on DBLP records, we could obtain the paper title information d_T to represent each paper. However, it is too limited to represent the paper only with the paper title. Thus Google Scholar is utilized for data supplementation, which will be discussed in Section 5.4.1. In our settings, each paper d is represented by the paper title d_T with its supplement d_S. Therefore, the relevance score is reformulated as the linear combination of both information,

$$\begin{aligned} f(q,d) &= \lambda_t p(q|\theta_{d_T}) + (1-\lambda_t) p(q|\theta_{d_S}) \\ &= \lambda_t \prod_{t \in q} p(t|\theta_{d_T})^{c(t,q)} + (1-\lambda_t) \prod_{t \in q} p(t|\theta_{d_S})^{c(t,q)}, \end{aligned} \tag{5.5}$$

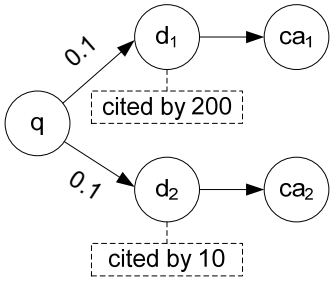


Figure 5.4: A query example with documents and authors.

where $p(t|\theta_{d_T})$ and $p(t|\theta_{d_S})$ are the smoothed term likelihoods of the paper title and its supplemental data respectively, and λ_t is the parameter to control the linear combination of both information.

5.2.3 Paper Importance

The language model described above calculates the relevance between a query and a document. Now we discuss the problem how to estimate the prior of the document. Generally, the document priors are assumed to be uniform, so $p(d)$ is ignored in previous studies. Let us see an example shown in Figure 5.4. There are two documents d_1 and d_2, associated with two authors ca_1 and ca_2 respectively. Suppose these two documents have the same relevance scores ($p(q|d_1) = p(q|d_2) = 0.1$) with respect to a query q. Given the above information, it is hard to answer the following two questions: (1) Which document is more reasonable to rank to the top? (2) Which author has the higher probability of being an expert given the query topic? Just imagine these two documents have different importance and d_1 is more important than d_2, we would obviously prefer to rank the more important one (d_1) at the top. Therefore, intuitively, it is more reasonable that ca_1 has the higher probability of being an expert than ca_2 on the query topic. The underlying assumption is that the more important the document is, the more

prior probability it is written by an expert. To the best of our knowledge, the document-based models [8, 48] currently do not take this factor into account.

We introduce a weight factor w_d to denote the importance of the document, which, theoretically, can be interpreted as being proportional to the document prior $p(d)$,

$$p(d) = \frac{w_d}{C_w} \propto w_d, \tag{5.6}$$

where C_w $(= \sum_{d \in D} w_d)$ is a constant normalization factor obtained by summing up all the document weights. The document priors are generally assumed to be uniform in previous studies. When the weight w_d is set to be uniform, we can see that this is exactly the existing methods with uniform document priors.

As shown in Figure 5.4, d_1 is cited by 200 documents, while d_2 is cited by 10 documents. When considering the citation number, the document d_1, which has the higher citation number, would be more important than d_2. For our model, the weight factor is estimated using the citation number, and transformed using the common logarithm function. We define two different methods to measure the weight as follows,

$$w_d = \begin{cases} 1, & (B1) \\ \log(10 + c_d), & (B2) \end{cases} \tag{5.7}$$

where c_d $(c_d \geq 0)$ is the citation number of the document d, and the constant 10 is used to guarantee the weight factor to be greater than 1. The citation numbers are obtained from Google Scholar. *B1* represents the baseline method with uniform weight, while *B2* is the weighting method that takes into account important factors of the documents.

5.2.4 Expertise Aggregation

So far, we have discussed how to estimate the relevance score and paper importance. Furthermore, we need to build the association between papers

and expert candidates. In scientific research, the publications of researchers could be viewed as the representative of their expertise [37]. Intuitively, their expertise could thus be deduced based on the overall aggregation of their publications.

Suppose document d_q is retrieved as one of the top-n related papers with respect to query q, then d_q should be quite relevant to the query q. Therefore, the candidate ca is assumed to be conditionally independent of the query q given a document d_q; that is

$$p(ca|d, q) = p(ca|d_q). \tag{5.8}$$

In our setting, it is reasonable to make the assumption that candidate ca has knowledge about the topic described in the document d if candidate ca is an author of document d. In the case of a paper with multiple authors, one author with many co-authors may have less association $p(ca|d)$ on average than a sole author. To account for this effect, we weight the association inversely according to the number of co-authors as follows. Suppose a document has n authors in total, we assume that each author has the same knowledge about the topics described in the document,

$$p(ca|d) = \begin{cases} \frac{1}{n_d}, & (ca \text{ is the author of } d) \\ 0, & (\text{otherwise}) \end{cases} \tag{5.9}$$

where n_d is the number of authors, and $p(ca|d)$ is used to measure the document-candidate association.

By substituting Eq. (5.5), Eq. (5.6) and Eq. (5.8) in Eq. (5.2), the final estimation of the joint probability would be

$$\begin{aligned} p(ca, q) &\overset{rank}{=} \sum_{d \in D} w_d f(q, d) p(ca|d_q) \\ &\overset{rank}{=} \sum_{d \in D} w_d \left(\lambda_t \prod_{t \in q} p(t|\theta_{d_T})^{c(t,q)} + (1 - \lambda_t) \prod_{t \in q} p(t|\theta_{d_S})^{c(t,q)} \right) p(ca|d_q), \end{aligned} \tag{5.10}$$

Table 5.1: Combination of different methods.

Model	w_d	**Refine**	**Meaning**
LM(bas)	B1[a]	N[c]	baseline model
LM(w)	B2[b]	N	weighted language model
LM(r)	B1	Y[d]	LM(bas) with graph-based regularization
LM(w+r)	B2	Y	LM(w) with graph-based regularization

[a] uniform weight ($w_d = 1$)

[b] common logarithm weight ($w_d = \log(10 + c_d)$)

[c] without graph-based regularization

[d] with graph-based regularization

where $\overset{rank}{=}$ means "equivalence for ranking the candidates." In this model, we incorporate the language model with paper importance, namely the weighted language model. When $w_d = 1$ and $\lambda_t = 1$, it can be regarded as the existing document-based model [8]. As shown in Table 5.1, *LM(bas)* represents the baseline model with uniform weight, while *LM(w)* is the method with the common logarithm weight. The expert lists are determined based on the joint probability. In Section 5.5.2, we evaluate the performance of the weighted language models with different weighting methods.

5.3 Graph-based Regularization

Under our proposed model, document retrieval is a key ingredient of the expert finding problem, and hence worth pursuing for identifying the most relevant papers along with the relevance scores. For a given query q, top-n related papers can be retrieved according to the relevance scores estimated by

the statistical language model. However, the statistical language model tends to be imperfect. In this section, we propose a novel and general framework to refine the relevance scores, so as to identify a set of the most relevant documents at the very top ranks of the final results.

5.3.1 Regularization Framework

With the advance of machine learning, graph-based models [40, 42, 109] have been widely and successively employed in information retrieval and data mining. A set of documents can be represented as a connected graph $G(V, E)$, where nodes V correspond to the n documents and edges E correspond to the explicit or implicit links between documents. Let $W \in \mathbb{R}^{n\times n}$ denote the weight matrix of the graph, where w_{ij} corresponds to the weight between d_i and d_j, and A is a diagonal matrix with entries $a_{ii} = \sum_j w_{ij}$.

Based on the graph, we propose a new regularization framework to refine the relevance scores. The underlying idea of the regularization framework is the global consistency on the graph: Similar documents are most likely to have similar ranking scores with respect to a query. In addition, the refined scores should be at least somewhat relevant to the initial relevance scores, which, in our framework, are constrained by a regularizer. Formally, we formulate the problem by minimizing a cost function $R(F, q, G)$ in a joint regularization framework similar to [151] as follows,

$$\begin{aligned} R(F, q, G) = \frac{1}{2}\sum_{i,j=1}^{n} w_{ij} \left\| \frac{f(d_i, q)}{\sqrt{a_{ii}}} - \frac{f(d_j, q)}{\sqrt{a_{jj}}} \right\|^2 \\ +\mu \sum_{i=1}^{n} \left\| f(d_i, q) - f^0(d_i, q) \right\|^2, \end{aligned} \tag{5.11}$$

where $\mu > 0$ is the regularization parameter, $f^0(d_i, q)$ is the initial relevance score of the document d_i against the query q, and $f(d_i, q)$ is the refined relevance score. We use F and F^0 to denote the refined and initial relevance

score vector, respectively. In the cost function, the first term defines the constraint to smooth the refined relevance scores on the graph, while the second term defines the constraint to fit the initial ranking scores, and the trade-off between each other can be controlled by the parameter μ.

The final ranking score vector is obtained by minimizing the cost function

$$F^* = arg \min_{F \in \mathbb{R}^{+n}} R(F, q, G). \tag{5.12}$$

Differentiating Eq. (5.11), we have

$$\frac{dR}{dF}\Big|_{F=F^*} = F^* - SF^* + \mu(F^* - F^0) = 0, \tag{5.13}$$

where $S = A^{-\frac{1}{2}} W A^{-\frac{1}{2}}$, then Eq. (5.13) can be transformed into

$$F^* - \frac{1}{1+\mu} SF^* - \frac{\mu}{1+\mu} F^0 = 0. \tag{5.14}$$

After simplifying, a closed-form solution can be derived,

$$\begin{aligned} F^* &= \mu_\beta (I - \mu_\alpha S)^{-1} F^0, \\ \mu_\alpha &= \frac{1}{1+\mu}, \text{ and } \mu_\beta = \frac{\mu}{1+\mu}, \end{aligned} \tag{5.15}$$

where I is an identity matrix, and S is a positive-semidefinite matrix. Note that μ_α ranges from 0 to 1, and $\mu_\alpha + \mu_\beta = 1$. In this chapter, we utilize the normalized Laplacian [151] to calculate matrix S. Details about how to calculate the matrix W and S will be introduced in Section 5.3.2. After obtaining the initial ranking scores F^0 and the matrix S, we can estimate the refined ranking scores F^* directly. With the graph-based regularization method, we develop another two models, *LM(r)* and *LM(w+r)*, for expert search as shown in Table 5.1.

5.3.2 Graph Construction

For the regularization framework, there are many ways to construct an adjacency graph and calculate the weight matrix W. To estimate the similarity

between two documents, we follow the method proposed in [73] and define the edge weight w_{ij} as follows

$$w_{ij} \stackrel{def}{=} \exp\left(-D\left(\theta_{d_i} \| \theta_{d_j}\right)\right), \tag{5.16}$$

where D is the Kullback-Leibler (KL) divergence [145, 148] based on the uniform Dirichlet-smoothed language model. To simplify the calculation, we only consider the paper title d_{Ti} for d_i, then $D\left(\theta_{d_i} \| \theta_{d_j}\right)$ can be reformulated to

$$D\left(\theta_{d_i} \| \theta_{d_j}\right) = \sum_{t \in d_{Ti}} p(t|d_{Ti}) \log \frac{p(t|d_{Ti})}{p(t|\theta_{d_j})} \tag{5.17}$$

where $p(t|d_T)$ is the likelihood $\frac{c(t,d_T)}{|d_T|}$ and $p(t|\theta_{d_j})$ is estimated by Eq.(5.4). Suppose d_i and d_j are the same document, its KL-divergence D will be 0, which results in $w_{ij} = 1$. On the other hand, if d_i and d_j are totally different, the KL-divergence D may be far less than 0, then w_{ij} will be close to 0.

In this chapter, we employ the widely used k-nearest neighbor (k-NN) graph, where each node is connected to its k nearest neighbors and the edges are weighed according to Eq. (5.16). The k-NN graph has been shown to be effective when $k = 10$ in [40]. After normalization, we can obtain the matrix $S = A^{-\frac{1}{2}} W A^{-\frac{1}{2}}$. This process is executed offline, and then we save the matrix S for our model.

5.3.3 Connections and Justifications

With respect to the difference between our regularization framework and other similar approaches in [95, 151, 156], our method is more general as it deals with a query-dependent function $f(d_i, q)$, while other methods are mainly used in query-independent settings, including semi-supervised classification and clustering.

As mentioned above, the balance between the initial ranking scores and the global consistency on the graph are tuned by the parameter μ_α, which

can be set between 0 and 1. If $\mu_\alpha \to 0$, i.e., $\mu \to +\infty$, then the regularization Eq. (5.11) puts almost all weight on the second term, and the objective function boils down to the initial *baseline*. By minimizing $R(F, q, G)$, we will obtain the results which best fit the initial ranking scores. When $\mu_\alpha \to 1$ ($\mu \to 0$), the regularization Eq. (5.11) puts almost all weight on the first term, and this objective function boils down to a variation of the PageRank-based model [149]. Minimizing $R(F, q, G)$ will give us the ranking scores which best fit the global consistency on the graph.

For the large-scale expert-finding problem, the matrix S is usually very large but sparse. However, the inverse matrix $(I - \mu_\alpha S)^{-1}$ will be very dense, which may require a huge space to save it. In order to balance the computation time and the storage space of the inverse matrix, we suggest to approximate the Eq. (5.15) in a specific subgraph. The subgraph (i.e., sub-matrix) $\hat{S}$ consists of the top-n documents according to the initial ranking scores $\hat{F}^0$. It can be found that the top ranking scores usually outnumber the very low ranking scores. Theoretically, if the ranking scores after n are close to 0, the following approximate solution is equivalent to Eq. (5.15):

$$\hat{F}^* = (I - \mu_\alpha \hat{S})^{-1} \hat{F}^0. \tag{5.18}$$

We eliminate the parameter μ_β in this equation as it does not change the ranking. According to the equation, it requires to calculate the inverse matrix $(I - \mu_\alpha \hat{S})^{-1}$ online. Fortunately, since the matrix is usually very sparse, it will reduce the time complexity of the matrix-inversion routine, which could be linear with the number of nonzero matrix elements. In our experiments, we extract the top 1,000-5,000 documents to approximate the sub-matrix.

```
<article mdate="2003-11-24" key="journals/cj/Fuhr92">
    <author>Norbert Fuhr</author>
    <title>Probabilistic Models in Information Retrieval.</title>
    <pages>243-255</pages>
    <year>1992</year>
    <volume>35</volume>
    <journal>Comput. J.</journal>
    <number>3</number>
    <url>db/journals/cj/cj35.html#Fuhr92</url>
</article>
```

Figure 5.5: A sample of the DBLP XML records.

5.4 Experimental Setup

In the following experiments we compare the expert finding models with different settings through an empirical evaluation. In this section we define the experimental setup, including the DBLP and topic collection, the assessments and evaluation metrics, while the evaluation results are presented in Section 5.5.

We have defined the following task: given a query and a set of expert candidates, the system has to retrieve a list of experts that have expertise in the given area. In the rest of this section, we introduce the DBLP and topic collection, the assessment and evaluation metrics.

5.4.1 DBLP Collection and Representation

The acquisition of a dataset populated with publications is an important aspect of finding experts from the bibliographic data which expertise can be derived. DBLP is a computer science bibliography website. As of November 2007, DBLP XML records contain over 955,000 articles, originally published in conferences, journals, books, etc., adding up to 414.5MB. One of the XML records is shown in Figure 5.5, which consists of several elements, such as "author" and "title." In total we gather more than 574,000 author names from

Figure 5.6: A snapshot for the search results of Google Scholar.

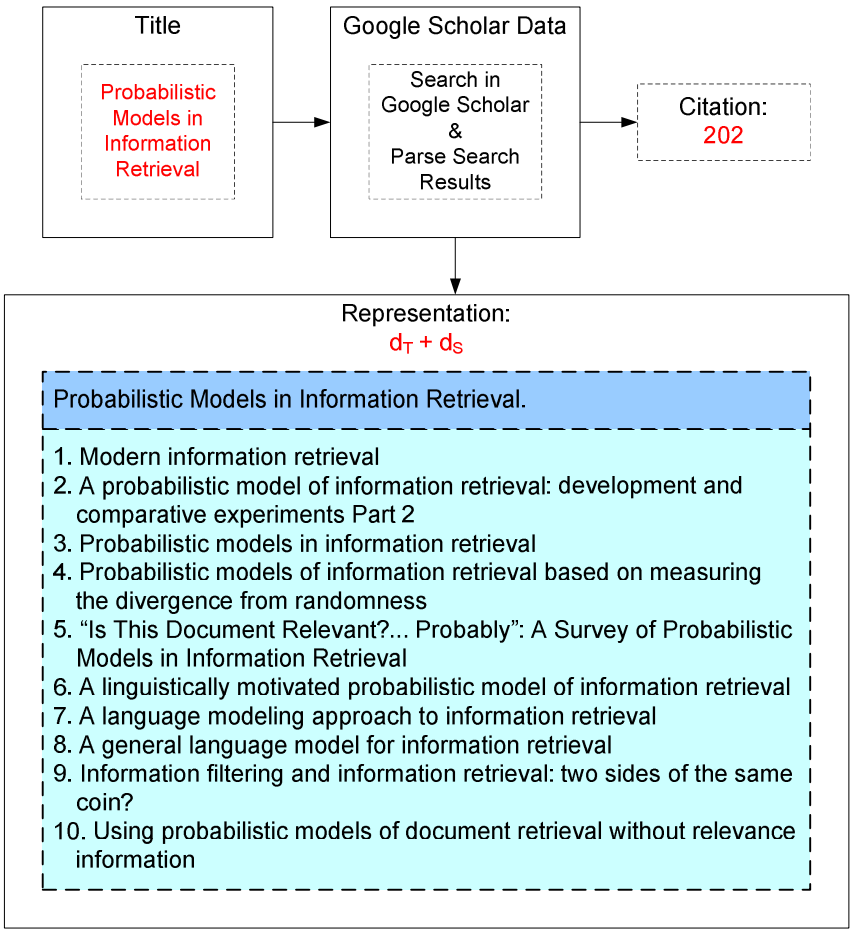


Figure 5.7: The representation of a document. After crawling and parsing the search results from Google Scholar, we combine the paper title d_T and the supplemental data d_S as the representation of a document.

DBLP XML records, each of which can be an expert candidate. Although DBLP is a good starting point for obtaining expert candidates and publications, several challenges exist due to its limitations. One limitation is that each DBLP record provides the paper title without the abstract and index terms. It is too limited to represent the paper based on the title; therefore, some more expanded information is required. Generally, the abstract and index terms are useful to represent the paper for estimating the probability of a query or topic given the paper.

To obtain the abstract and index terms for each DBLP record, one natural way is to fetch them automatically from digital libraries such as ACM[1], IEEE[2], Springer[3], etc. We note, however, that it is very difficulty to obtain the complete metadata, i.e., the abstracts and index terms of publications, for all the DBLP records. In order to obtain a complete, consistent and effective representation, Google Scholar is utilized for data supplementation. For each paper d, we use the title as the query to search in Google Scholar and select the top 10 returned records which are considered most relevant to the query title. Figure 5.6 illustrates the search results for the paper "probabilistic models in information retrieval." As shown in this figure, the third returned record corresponds to the paper itself, and it is cited by 202 other papers. Next, these returned records d_S are parsed and combined with the paper title d_T as the representation of paper d. This process, as depicted in Figure 5.7, is done automatically by a crawler and a parser. Moreover, the citation number for each publication d is obtained at the same time. The metadata (HTML pages), up to 20GB, is crawled from Google Scholar in 2007. As shown in Table 5.2, the total number of valid papers after this process is $925,293$, the number of authors is $574,369$, and the number of terms is 308,651.

[1] http://portal.acm.org/

[2] http://ieeexplore.ieee.org/

[3] http://www.springer.com

Table 5.2: Statistics of the DBLP collection.

Property	#of entities
Number of papers	925,293
Number of authors	574,369
Number of terms	308,651

5.4.2 Assessments

To evaluate the quality of retrieved experts, we manually created the ground truth through the method of pooled relevance judgments with human assessment efforts. For each query, the top authors from the computer science bibliography search engines (such as CiteSeer[4], Libra[5], and Rexa[6]) and the committees of the top conferences in the topic were taken to construct the pool. Then some researchers were asked to assess each of the recommended candidates with respect to the query. To help them in their task, those researchers were presented with publications and a description relating to each author. They could access and find additional content directly on a search engine when needed.

Such a benchmark dataset with expert lists (for expert finding) has been collected in Tsinghua University [150]. It contains 7 query topics and creates 7 expert lists. Their assessments were carried out mainly in terms of the number of top conference/journal papers an expert candidate has published, the number of related publications for the given query, and what distinguished awards he/she has been awarded. There are four grade scores (3, 2, 1, and 0) which were assigned respectively to represent top expert, expert, marginal expert, and not expert. Finally, the judgment scores (at levels 3 and 2) were

[4]http://citeseer.ist.psu.edu/
[5]http://libra.msra.cn/
[6]http://rexa.info/

averaged to construct the final ground truth. We extended this data set to contain 17 query topics and 17 expert lists. Table 5.3 shows the details of the benchmark dataset.

5.4.3 Evaluation Metrics

For the evaluation of the task, three different metrics are adopted to measure the performance of our proposed models, including precision at rank n (P@n), mean average precision (MAP), bpref [19].

Precision at rank n (P@n) P@n measures the fraction of the top-n retrieved results that are relevant experts for the given query, which is defined as

$$P@n = \frac{\#\text{ relevant experts in top } n \text{ results}}{n}.$$

R-precision (R-prec) is defined as the precision at rank R where R is the number of relevant candidates for the given query.

Mean Average Precision (MAP) Average precision (AP) emphasizes returning more relevant documents earlier. For a single query, AP is defined as the average of the P@n values for all relevant documents:

$$AP = \frac{\sum_{n=1}^{N}(P@n * \text{rel}(n))}{R},$$

where n is the rank, N the number retrieved, and rel(n) is a binary function indicating the relevance of a given rank. MAP is the mean value of the average precisions computed for all the queries.

Bpref Beside the measurement of precisions, Bpref [19] is a good score function that evaluates the performance from a different view, i.e., the number of non-relevant candidates. It is formulated as

$$\text{bpref} = \frac{1}{R}\sum_{r=1}^{N}(1 - \frac{\#n \text{ ranked higher than } r}{R}),$$

Table 5.3: Benchmark dataset of 17 queries.

Topic	**#Expert**
Boosting	56
Information Extraction	20
Intelligent Agents	29
Machine Learning	42
Natural Language Processing	43
Planning	34
Semantic Web	45
Support Vector Machine	31
Ontology Alignment	55
Probabilistic Relevance Model	13
Information Retrieval	23
Language Model For Information Retrieval	12
Face Recognition	21
Semi Supervised Learning	21
Reinforcement Learning	17
Privacy Preservation	17
Kernel Methods	22

where r is a relevant candidate and n is a member of the first R candidates judged non-relevant as retrieved by the system.

In our experiments, we report the results of P@5, P@10, P@20, R-prec, MAP, and bpref.

5.5 Experimental Results

The presentation of the experimental results is organized in the following four subsections. First we evaluate the baseline model for expert finding, and compare the performance of two different representations for the DBLP collection. Then the experiments are performed to compare the weighted language models with the paper importance in Section 5.5.2. In Section 5.5.3, we examine the effectiveness of the graph-based regularization in our model. Finally, the detailed results are discussed in Section 5.5.4. The experimental results shown in this section are the average results.

5.5.1 Preliminary Experiments

In order to compare the performance of different representations, we set up and index two corpora for evaluation. One corpus ("Title") is collected only using the paper title, while the other corpus ("Title+GS") is built based on the combination of the paper title and its supplemental representation using Google Scholar. Different representations result in different paper relevance scores for a given query. As shown in Eq.(5.5), the relevance score is controlled by the parameter λ_t. When $\lambda_t = 1$, the score is determined by $p(q|\theta_{d_T})$, and it means the paper is represented only using the publication title d_T. When $\lambda_t \in (0, 1)$, the score is determined by both $p(q|\theta_{d_T})$ and $p(q|\theta_{d_S})$, which means the paper is represented based on the combination of the title and its supplemental data d_S.

Table 5.4: Experimental results with different representations (%).

	P@5	P@10	P@20	R-prec	MAP	bpref
"Title"	61.18	51.18	44.71	40.30	27.27	33.20
"Title+GS"	72.94	64.12	47.94	43.98	33.06	38.16

We perform the preliminary experiments using the basic language model *LM(bas)* with two parameters: $\lambda_t = 1$ ("Title") and $\lambda_t = 0.5$ ("Title+GS"). The comparison of the results is reported in Table 5.4. It is quite clear that the results of "Title+GS" are much better than those of "Title" in all the metrics from P@5 to bpref, especially the very top precision. For the precision P@5, "Title+GS" achieves 72.94%, which is about 11.7% higher than the results of "Title." According to Table 5.4, it is more effective to represent publications using Google Scholar for data supplementation. In all these experiments, we retrieve the top 1,000 most relevant papers and set $\nu = 10$ for Dirichlet prior smoothing. In the following parts, we set λ_t to be 0.5 with "Title+GS."

5.5.2 Language Models with Paper Importance

In this subsection, the effect of paper importance is studied and evaluated. We compare the performance of the weighted language models with two different weighting methods. As mentioned in Table 5.1, *LM(bas)* represents the baseline method with uniform weight $w_d = 1$, while *LM(w)* is the method with the common logarithm weight $w_d = \log(10 + n_d)$. Table 5.5 shows the results for the different methods with "Title" and "Title+GS", respectively.

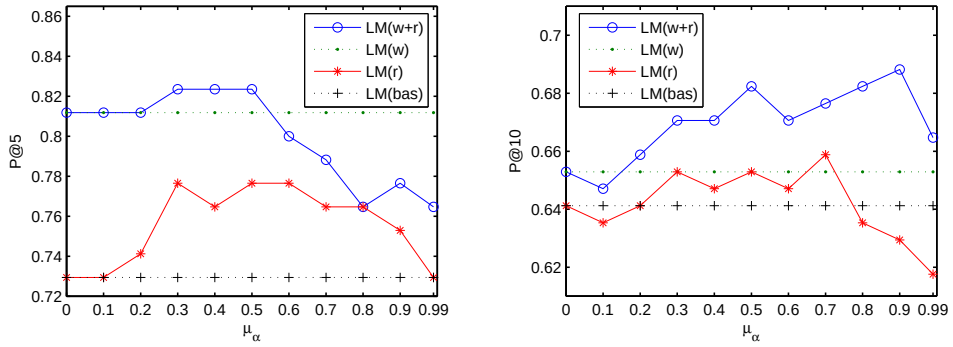
First, we inspect the absolute performance of the methods. For the precision P@5, the basic language model *LM(bas)* achieves 61.18% and 72.94% respectively for "Title" and "Title+GS", while the weighted language models *LM(w)* can enhance the precision significantly to 72.94% and 81.18% respec-

Table 5.5: Evaluation results of language models using different weighting methods (%). Best scores are in boldface.

"Title"	P@5	P@10	P@20	R-prec	MAP	bpref
LM(bas)	61.18	51.18	44.71	40.30	27.27	33.20
LM(w)	72.94	60.59	48.53	43.22	31.91	36.79
	+19.2	+18.4	+8.55	+7.25	+17.0	+10.8
"Title+GS"	P@5	P@10	P@20	R-prec	MAP	bpref
LM(bas)	72.94	64.12	47.94	43.98	33.06	38.16
LM(w)	**81.18**	**65.29**	**53.24**	**47.93**	**37.10**	**41.60**
	+11.3	+1.84	+11.0	+8.98	+12.2	+9.01

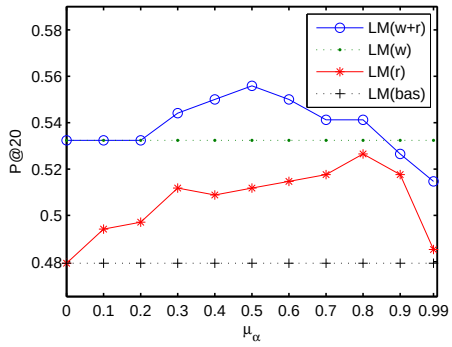
tively. With reference to the MAP, the *LM(w)* has a 17.0 percent improvement over the *LM(bas)* for "Title", and a 12.2 percent improvement over the *LM(bas)* for "Title+GS." When looking at the overall performance of the various models, we observe that the weighted language model *LM(w)* outperforms the basic language model *LM(bas)* on all the metrics from P@5 to bpref, not only on "Title" corpus but also on "Title+GS" corpus.

According to the experimental results, we can argue that it is very important to consider the prior probability of the document in the model. By way of taking into account the paper importance and the paper relevance simultaneously, the weighted language model performs very well and achieves much better performance than the basic language model with the uniform weight. Another interesting observation is that the results of "Title+GS" are better than those of "Title", which reconfirms the effective representation of "Title+GS."

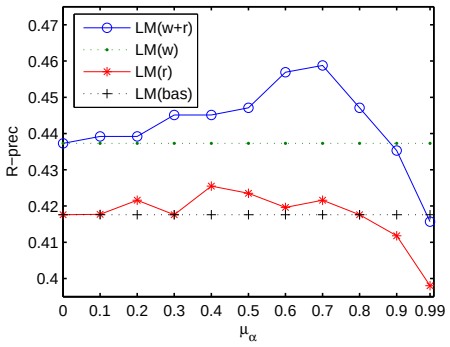


(a) P@5

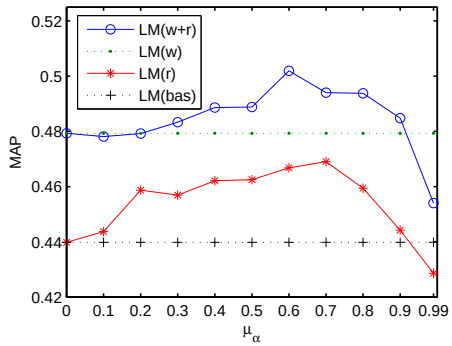
(b) P@10



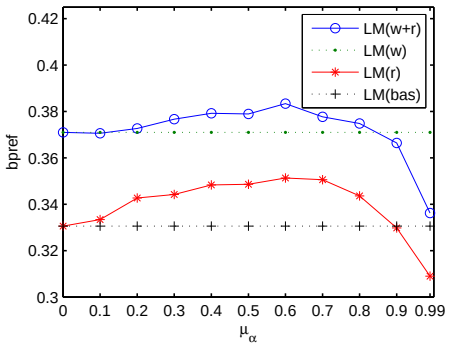
(c) P@20



(d) R-prec



(e) MAP



(f) bpref

Figure 5.8: The effect of varying the parameter μ_α by comparing four different models, including the basic language model *LM(bas)*, basic language model with graph-based regularization *LM(r)*, weighted language model *LM(w)*, and its extension with graph-based regularization *LM(w+r)*.

5.5.3 Effect of Graph-based Regularization

We have shown the effectiveness and improvement of our weighted language models in previous subsections. We now consider the question whether the graph-based regularization framework can further boost the performance of both *LM(bas)* and *LM(w)* models. The objective of the regularization framework is to refine the initial relevance scores, so as to identify the most relevant papers as well as the relevant experts for a given query. In the unified regularization framework, the parameter μ_α is used to control the balance between the global consistency and the initial ranking scores. We study and evaluate the effect of the parameter μ_α by setting different values in this subsection. If μ_α is set to be 0, the regularization framework boils down to the initial *baseline*. When $\mu_\alpha \rightarrow 1$, it discards the initial ranking scores, and only takes into account the global consistency on the graph according to Eq. (5.11).

In order to evaluate the robustness of the proposed regularization framework, we set ten different values (from 0.1 to 0.99) for μ_α, and examine the corresponding performance. We compare four different models, including the basic language model *LM(bas)*, basic langauge model with graph-based regularization *LM(r)*, weighted language model *LM(w)*, and its extension with graph-based regularization *LM(w+r)*. The experimental results for different parameter μ_α are illustrates in Figure 5.8, and all of these experiments are performed on "Title+GS." In this figure, the solid curves denote the results of *LM(r)* and *LM(w+r)*, which vary with the parameter μ_α. The dashed lines denote the *baseline* results of *LM(bas)* and *LM(w)*. When $\mu_\alpha = 0$, the results of *LM(r)* and *LM(w+r)* are the same as those of *LM(bas)* and *LM(w)*, respectively.

Let us look at the MAP metric as shown in Figure 5.8(e) in detail. Compared to the *baseline LM(bas)*, *LM(r)* can boost its performance when incorporating the global consistency in the framework ($\mu_\alpha >= 0.2$). With the

increase of μ_α, the performance becomes better. Meanwhile, similar improvements occur in the results of *LM(w+r)* by comparing with *LM(w)*. When the parameter μ_α is equal to 0.99, the performance of both *LM(r)* and *LM(w+r)* becomes worse than that of the initial *LM(bas)* and *LM(w)* methods. This is because of the overweighted global consistency in the framework if it puts too much weight on the first term ($\mu_\alpha \rightarrow 1$). In terms of the comparison using other metrics, it has shown similar trends about the parameter μ_α. The experimental results demonstrate the effectiveness of the proposed graph-based regularization framework. Moreover, the regularization framework is relatively robust and may achieve the best results when the parameter μ_α is set to be 0.5-0.7. The parameter μ_α used in Section 5.5.4 is therefore set to be 0.5.

5.5.4 Comparison and Detailed Results

We show the comparison of different models in Table 5.6. The first part shows the absolute precisions of those models, and the second part illustrates the percentage of relative improvements. A quick scan of the first part, accompanying with Figure 5.8, reveals that *LM(w+r)* achieves the best performance from P@5 to bpref. When looking at the top two lines of the lower part, we can see that *LM(r)* improves over the baseline *LM(bas)* consistently for all the metrics. The relative improvements of *LM(w+r) / LM(w)* are less than those of *LM(r) / LM(bas)* in most metrics except P@10. The most important observation is that graph-based regularization can further boost the performance of the weighted language models. In terms of the relative improvements over the baseline *LM(bas)*, we can see that *LM(r)* improves over *LM(bas)* by up to 6.75% for P@20, *LM(w)* over *LM(bas)* by up to 12.22% for MAP, and *LM(w+r)* over *LM(bas)* by up to 15.95% for P@20. In general, *LM(w+r)* achieves the best performance. According to the experimental re-

sults, we can argue that it is very essential and promising to consider the weighted language model with graph-based regularization for expert finding.

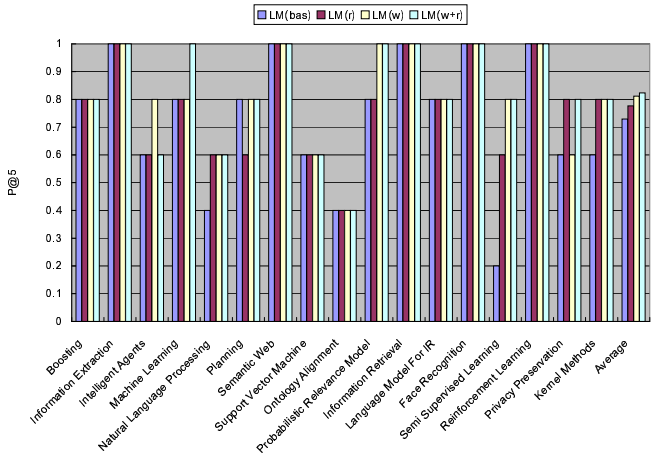
To gain a better insight into the details of the results, we compare with the experimental results of these four methods on each query. Figure 5.9 shows the detailed results of these methods on 17 queries. From the detailed experimental results, we can see that our *LM(w+r)* method outperforms the baseline as well as other two methods in most cases. On the other hand, there are few cases that *LM(w+r)* does not make an improvement, for example, the query "machine learning", "support vector machine" and "reinforcement learning" in Figure 5.9(b). However, the overall performance (MAP and bpref) of those three queries, as shown in Figure 5.9(e) and Figure 5.9(f), are better than that of the baseline. We also see that the performance (MAP) in some queries, for instance, the query "ontology alignment", is not good enough. This is because most of the data (papers/authors) corresponding to those queries are uncovered in the collection. From the figure, we can observe the improvement of our *LM(w+r)* method is more consistent and promising when compared to the baseline and the other two methods.

5.6 Summary

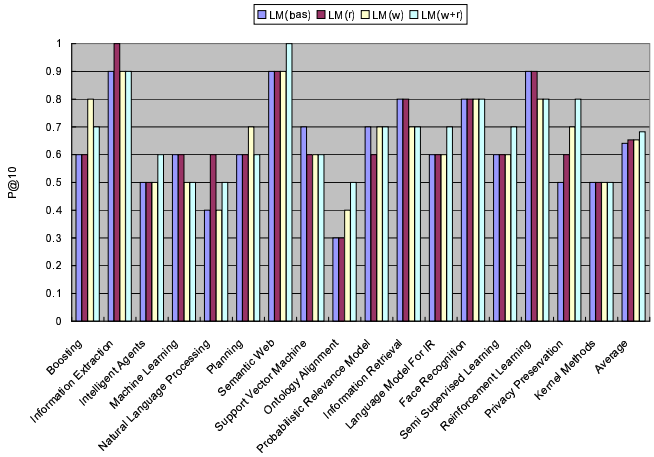
In this chapter we present the weighted language model for expert finding, whose basic idea is to aggregate the expertise of a candidate from the associated documents. Our proposed model first retrieves the most relevant documents with respect to a query, and then takes into account not only the relevance scores, but also the importance of the documents simultaneously. Furthermore, we investigate and integrate the graph-based regularization method to enhance our model, which leads to a further improvement by leveraging the global consistency over the graph to refine the relevance scores. We have conducted an extensive set of experiments on a benchmark dataset

Table 5.6: Comparison of different methods (%). The percentages of relative improvements are shown in the lower part.

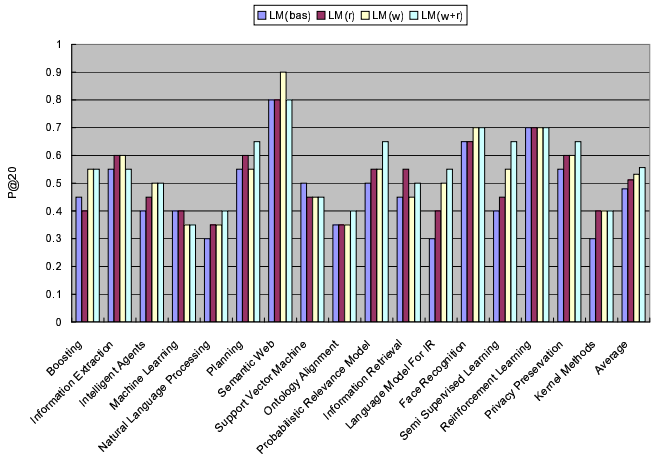
Method	P@5	P@10	P@20	R-prec	MAP	bpref
LM(bas)	72.94	64.12	47.94	43.98	33.06	38.16
LM(r) ($\mu_\alpha = 0.5$)	77.65	65.29	51.18	46.25	34.86	39.97
LM(w)	81.18	65.29	53.24	47.93	37.10	41.60
LM(w+r) ($\mu_\alpha = 0.5$)	82.35	68.24	55.59	48.88	37.89	42.60
LM(r) / LM(bas)	+6.45%	+1.83%	+6.75%	+5.15%	+5.42%	+4.75%
LM(w+r) / LM(w)	+1.45%	+4.50%	+4.42%	+1.97%	+2.13%	+2.40%
LM(w) / LM(bas)	+11.29%	+1.84%	+11.04%	+8.98%	+12.22%	+9.01%
LM(w+r) / LM(bas)	**+12.90%**	**+6.42%**	**+15.95%**	**+11.13%**	**+14.61%**	**+11.63%**



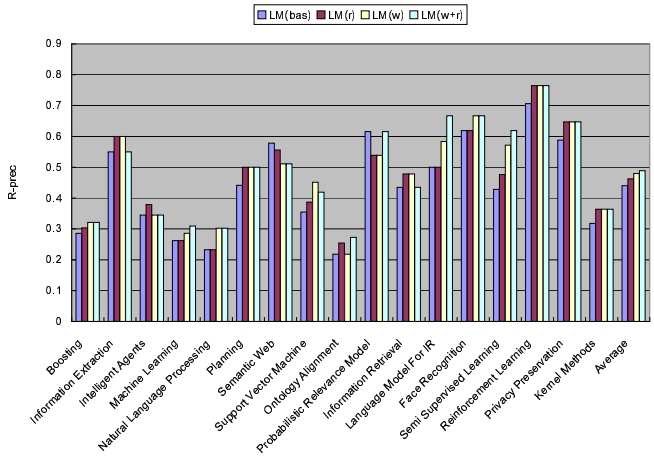
(a) P@5



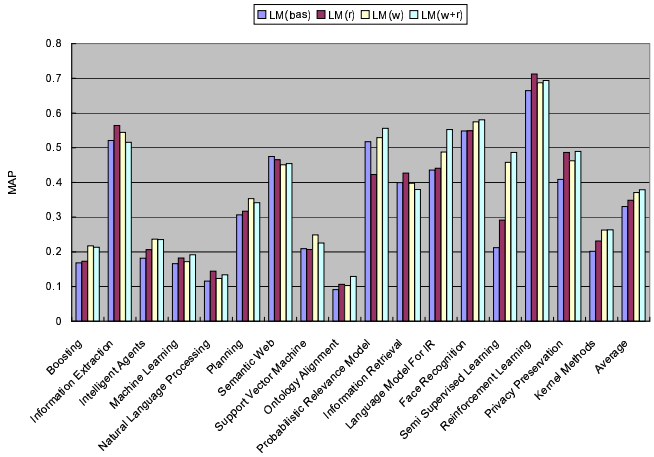
(b) P@10



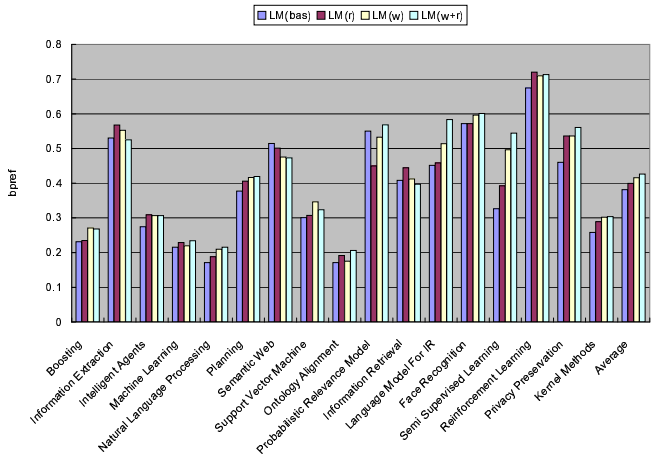
(c) P@20



(d) R-prec



(e) MAP



(f) bpref

Figure 5.9: Illustration and comparison of the experimental results on each query for four different methods.

for evaluating the performance of a number of algorithms with different settings. The promising experimental results validate the effectiveness of our weighted language model, and demonstrate that it can be further improved with the graph-based regularization method.

□ **End of chapter.**

Chapter 6

Enhancing Expertise Retrieval

Motivated by the observation that communities could provide valuable insight and distinctive information, we investigate and develop two community-aware strategies to enhance the expertise retrieval. We first propose a new smoothing method using the community context for statistical language model, which is employed to identify the most relevant documents so as to reflect the expertise retrieval in the document-based model. Furthermore, we propose a query-sensitive AuthorRank to model the authors' authorities based on the community coauthorship networks, and develop an adaptive ranking refinement method to enhance the expertise retrieval. Experimental results demonstrate the effectiveness and robustness of both community-aware strategies.

6.1 Motivation

Expertise retrieval has received increased interests in recent years, whose task is to suggest people with relevant expertise to the topic of interest. One of the state-of-the-art approaches [8, 37] is the document-based model using a statistical language model to rank experts. However, one of the issues is that previous algorithms mainly consider the documents associated with the experts, while ignoring the community information that is affiliated

with the documents and the experts. Actually, in addition to the associated documents, there is much other information that can be included, such as the community context information and the community social information. Therefore, how to utilize the community-based information to enhance the expertise retrieval becomes an interesting and challenging problem.

Given a set of documents and their authors, it is possible and often desirable to discover and infer the community information, in which contains a number of documents and authors for each community. Some existing studies have been conducted about how to discover community [78, 152], but this is not the focus of our work: here we focus on the problem of enhancing expertise retrieval with the community information, and suppose the community information is already existed. As our approach is to deal with the expert-finding task in a real-world academic domain, it is reasonable to assume the academic communities have been formed automatically in the form of conferences and journals, in which the researchers publish their papers, exchange their ideas, and coauthor with each other.

An illustrated graph with two communities is sketched in Figure 6.1. There are five documents associated with five authors. The edge between a document and an author means the document is written by the author. We assume each document d_i can only belong to one community C_k, and each author a_j of the document is affiliated with the corresponding community C_k. In this example, d_1 and d_2 belong to the community C_1, and meanwhile d_3, d_4 and d_5 form the community C_2. For the authors of the documents, a_1, a_2 and a_3 are affiliated with the community C_1, and a_3, a_4 and a_5 with the community C_2, so a single author may belong to multiple communities. There is a pair of distributions for each community: one over documents and one over authors. With such community-based information, the community can be represented from two different perspectives to obtain the community con-

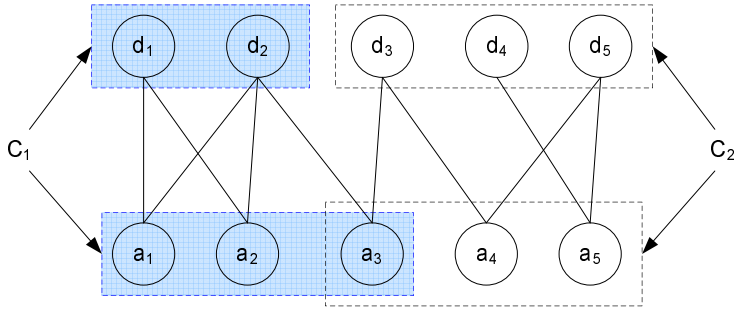


Figure 6.1: An example graph with two communities

text (text information) based on the papers and the community coauthorship network based on the authors.

In this chapter, we propose two community-aware strategies to enhance the expertise retrieval. The first one is the community-based smoothing method for statistical language model, which is employed to identify the most relevant documents so as to reflect the expertise retrieval in the document-based model. The smoothing method is an important characteristic of the language model for computing the relevance score. In previous approaches [8, 37], the document language model is smoothed by the whole collection language model, which smooths each word equally in all the documents while ignoring their different community information. However, we argue that the community context provide more valuable and distinctive information for the document than the whole collection. For example, as shown in Figure 6.1, suppose C_1 denote a "machine learning" community, and C_2 denotes a "information retrieval" community. Thus it is likely to contain a higher proportion of words related to "machine learning" in the context of C_1 than the whole collection, and meanwhile there would be a higher proportion of words related to "information retrieval" in the context of C_2 than the whole collection. This observation motivates us to conduct the novel smoothing method using the

community context.

Moreover, the second strategy is developed to boost the document-based model using the community-sensitive authorities. More specifically, we propose a query-sensitive AuthorRank to model the authors' authorities based on the coauthorship networks, and develop an adaptive ranking refinement method to aggregate the ranking results. Intuitively, experts usually have high authorities in some communities, which reflect their general and high-level expertise in some aspects. In contrast, the document-based model reflects more specific and detailed aspects for expertise retrieval, as it measures the contribution of each document individually. From this point of view, the community-sensitive authorities should be taken into consideration along with the document-based model for expertise retrieval, which is referred to as the enhanced model. To illustrate our methodology, we apply the proposed methods to the expert finding task using the DBLP bibliography data [1]. Experimental results demonstrate the effectiveness and robustness of the community-aware strategies. Moreover, the improvements made in the enhanced model are significant and consistent.

In this chapter, our major contributions are: (1) the investigation of the smoothing method using community context instead of the whole collection to enhance the language model for the document-based model; (2) the introduction of the community-sensitive AuthorRank for determining the query-sensitive authorities for experts; and (3) an adaptive ranking refinement strategy to aggregate the ranking results of both document-based model and community-sensitive AuthorRank, which leads to a significant improvement over the baseline method.

The rest of this chapter is organized as follows. Section 6.2 describes the preliminaries of the expertise modeling. Section 6.3 presents the document-based models smoothed using community context. Section 6.4 describes the

enhanced models with community-aware authorities. Section 6.5 defines the experimental setup and reports the experimental results. Section 6.6 summarizes this chapter.

6.2 Preliminaries

Suppose $A = \{a_1, a_2, ..., a_M\}$ is the set of expert candidates (i.e., authors) to be retrieved. Let $D = \{d_1, d_2, ..., d_N\}$ denote a collection of supporting documents, where d_i is a paper authored by one or several candidates. Let $\mathbb{C} = \{C_1, C_2, ..., C_K\}$ denote the collection of corresponding communities, where C_k consists of a set of papers and their associated authors. As illustrated in Figure 6.1, the relationships between authors, documents and communities can be represented by the tuple $< a_i, d_j, C_k >$, signifying that author i has a paper j that is published in the community k. Note that each paper exclusively belongs to one community, while an author may belong to multiple communities.

For a given query q, the problem of identifying experts is formulated using a generative probabilistic model, i.e., what is the probability of a candidate a_i being an expert given the query topic q? Using Bayes' theorem, the probability can be formulated as follows:

$$p(a_i|q) = \frac{p(a_i, q)}{p(q)} \propto p(a_i, q), \tag{6.1}$$

where $p(a_i, q)$ is the joint probability of a candidate and a query, $p(q)$ is the probability of the query. The probability $p(q)$ is a constant, so it can be ignored for ranking purposes. To derive the probability $p(a_i|q)$, it is equivalent to estimate the joint probability $p(a_i, q)$.

A number of methods have been proposed to estimate the probability $p(a_i, q)$. One successful method, proposed by Deng et al. [37], decomposes the joint probability into the product over the supporting documents using a

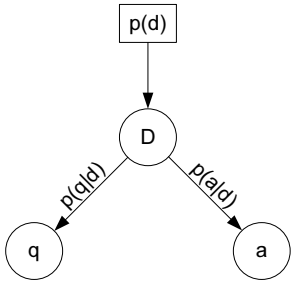


Figure 6.2: The document-based model for expertise retrieval.

generative probabilistic model. The basic idea is to model the expertise of an expert based on the relevance and importance of the associated documents. As shown in Figure 6.2, the supporting documents D act as a "bridge" to connect the query q with the candidate a. We follow this approach (document-based model) to estimate the probability as

$$\begin{aligned} p_d(a_i, q) &= \sum_{d_j \in D} p(d_j) p(a_i, q|d_j) \\ &= \sum_{d_j \in D} p(d_j) p(q|d_j) p(a_i|d_j, q) \\ &= \sum_{d_j \in D} p(d_j) p(q|\theta_{d_j}) p(a_i|d_j), \end{aligned} \tag{6.2}$$

where $p(d_j)$ is the prior probability of a document, $p(q|d_j)$ means the relevance between q and d_j, and $p(a_i|d_j, q)$ represents the association between the candidates and the documents for a given query. In this equation, we assume the candidate a is conditionally independent of the query q given a document d; that is $p(a_i|d_j, q) = p(a_i|d_j)$.

6.3 Document-based Models with Community-Aware Smoothing

As stated before, the task of the expertise retrieval is to retrieve a list of experts that have expertise for the given query. In this section, we describe

the document-based models with community-aware smoothing strategy for the real-world academic domain.

6.3.1 Statistical Language Model

In the document-based model, one of the key challenges is to compute the relevance between a query and documents. In recent years, statistical language model has been widely used in the application of information retrieval [60, 108, 145, 148]. To determine the probability of a query given a document, we infer a document language model θ_d for each document. The relevance score of document d with respect to query q is then defined as the conditional probability $p(q|\theta_d)$. Suppose $q = t_1...t_m$ and each word t is generated independently, the relevance score would be,

$$p(q|\theta_d) = \prod_{t_i \in q} p(t_i|\theta_d), \tag{6.3}$$

where $p(t|\theta_d)$ represents the maximum likelihood estimator of the word in a document d.

With such a model, the retrieval problem is reduced to the problem of estimating $p(t_i|\theta_d)$. In order to assign nonzero probabilities to unseen words, it is important to incorporate the smoothing methods in estimating the document language model. One popular way to smooth the maximum likelihood estimator is the Jelinek-Mercer smoothing method with the collection language model:

$$p(t|\theta_d) = (1 - \lambda)\frac{n(t, d)}{|d|} + \lambda p(t|G), \tag{6.4}$$

where λ is the parameter to control the amount of smoothing, $n(t, d)$ is the count of word t in the document d, $|d|$ is the number of the words in d, and $p(t|G)$ is the collection language model. The collection language model can be estimated by normalizing the count of words in the entire collection, which

can be defined as

$$p(t|G) = \frac{\sum_{d_j \in G} n(t, d_j)}{\sum_{d_j \in G} |d_j|}. \tag{6.5}$$

Accordingly, we can define the collection-smoothed language model as

$$p(q|\theta_d) = \prod_{t_i \in q} \left((1 - \lambda) \frac{n(t_i, d)}{|d|} + \lambda p(t_i|G) \right). \tag{6.6}$$

6.3.2 Smoothing Using Community Context

Now we investigate how to use the community information to enhance the language model described above. In this subsection, a novel smoothing method is proposed for the document language model by leveraging the community-aware information to determine the probability $p(q|\theta_d)$.

Suppose the community information is already existed for each document. For example, a conference or journal, which contains a set of publications, can be treated as a community. Figure 6.3 illustrates the relationships between the documents, the communities and the whole collection. There are three-level representations for the language model: the variable θ_d denotes the low-level document representation, sampled once per document; the variable C_d denotes the middle-level community representation, consisted of a set of documents including d; finally, the variable G denotes the high-level collection representation, consisted of all the documents.

According to the traditional language model, each word is smoothed by the same collection language model, which would be treated equally despite of their different community information. However, the community provides valuable insight and distinctive information for its documents rather. Because a document will somewhat share much more common information with its community rather than the whole collection. Moreover, each community may have its own distinctive characteristics, which are different from other communities. Therefore, it would be more reasonable to employ the distinctive community language model, instead of the whole collection based

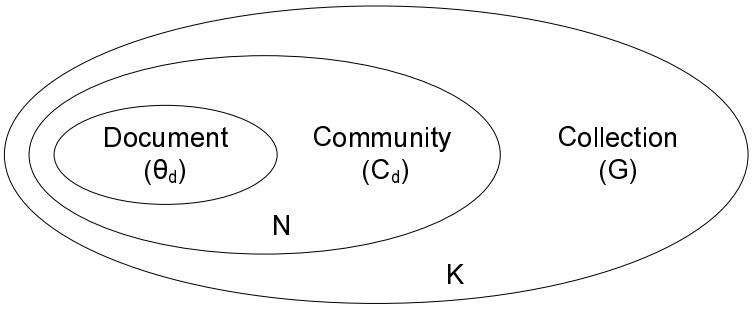


Figure 6.3: A graph representation of the relationships between documents, communities and the entire collection.

smoothing, to smooth different document models. The community language model is defined as

$$p(t|C_d) = \frac{\sum_{d_j \in C_d} n(t, d_j)}{\sum_{d_j \in C_d} |d_j|}. \tag{6.7}$$

For two documents that belong to two different communities, we can define two distinctive community language models, instead of the same collection language model, to smooth the document language model. The community-smoothed language model is obtained by substituting $p(t|C_d)$ for $p(t|G)$ into Eq. (6.6)

$$\hat{p}(q|\theta_d) = \prod_{t_i \in q} \left((1-\lambda) \frac{n(t_i, d)}{|d|} + \lambda p(t_i|C_d) \right). \tag{6.8}$$

Note here the document d belongs to the community C_d.

6.3.3 Determining Other Probabilities

We have described two language models for calculating the probability $p(q|d)$, now we proceed to introduce the estimation of the other probabilities $p(d)$ and $p(a|d)$. Generally, the document prior $p(d)$ is assumed to be uniform. In addition, $p(d)$ is interpreted as the document importance in [37], which is estimated based on the citation of the document. We briefly define these two

Table 6.1: Combination of different methods.

Model	**w**[a]	**c**[b]	**E**[c]	**Remarks**
Document-based models				
$DM(b)$	B1	0	0	baseline
$DM(bc)$	B1	1	0	community-based smoothing
$DM(w)$	B2	0	0	weighted model
$DM(wc)$	B2	1	0	community-based smoothing
Enhanced models				
$EDM(b)$	B1	0	1	enhanced $DM(b)$
$EDM(bc)$	B1	1	1	enhanced $DM(bc)$
$EDM(w)$	B2	0	1	enhanced $DM(w)$
$EDM(wc)$	B2	1	1	enhanced $DM(wc)$

[a] uniform weight (B1) or common logarithm weight (B2)

[b] smoothing using the community (1) or collection (0)

[c] enhancing with community-aware authorities (1) or no enhancement (0)

weight methods as follows,

$$p(d) \propto \begin{cases} 1, & (B1) \\ \log\left(10 + N_c(d)\right), & (B2) \end{cases} \tag{6.9}$$

where $N_c(d)$ is the citation of d, and the constant 10 is used to guarantee the weight to be greater than 1. The probability $p(a|d)$ indicates the association between papers and authors. One simple way is to define the probability inversely according to the number of authors. Suppose a document has multiple authors in total, each author is assumed to have the same knowledge about the topics described in the document,

$$p(a|d) = \begin{cases} \frac{1}{N_a(d)}, & (a \text{ is the author of } d) \\ 0, & (\text{otherwise}) \end{cases} \tag{6.10}$$

where $N_a(d)$ is the number of authors for the document.

So far, there are two language models, i.e., as Eq. (6.6) and Eq. (6.8), for calculating $p(q|\theta_{d_j})$, and two methods in Eq. (6.9) for computing $p(d)$ as well. By considering each method and substituting into Eq. (6.2) separately, four different models can be combined as shown in the upper part of Table 6.1. We evaluate and compare the performance of these document-based models in Section 6.5.3.

6.4 Enhanced Models with Community-Aware Authorities

In the academic domain, researchers in similar fields are most likely to form a community, and to publish relevant articles in the community. Motivated by the observation that experts usually have high authorities in some communities, we develop and investigate the query-sensitive authorities with an

adaptive ranking refinement strategy, so as to enhance the expertise retrieval models.

6.4.1 Discovering Authorities in a Community

In a community, the authors' relationships can be described using a coauthorship network. Coauthorship network is an important category of social networks, and has been used extensively to determine the structure of scientific collaborations [100]. We consider the weighted directed graph to model the coauthorship network in which each edge represents a coauthorship relationship. If any two authors coauthored a paper, an edge with a weight is created. Let us take the community C_1 in the Figure 6.1 as an example. Authors a_1 and a_2 coauthored paper d_1, and a_1, a_2 and a_3 coauthored paper d_2. So a_1, a_2 and a_3 would be connected with each other.

To quantify the edge weight, the coauthorship frequency is proposed in [83], which consists of the sum of all values for all papers coauthored by a_i and a_j,

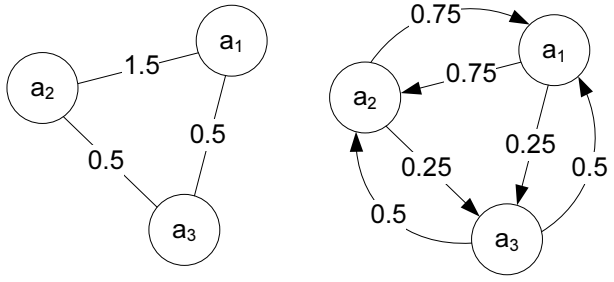
$$f_{ij} = \sum_{k=1}^{N} \frac{\delta_i^k \delta_j^k}{n_k - 1}, \tag{6.11}$$

where $\delta_i^k = 1$ if a_i is one of the authors of the paper d_k, otherwise $\delta_i^k = 0$, and n_k is the number of authors in paper d_k. This gives more weight to authors who co-publish more papers together. For the example above, the graph with the coauthorship frequency is illustrated in Figure 6.4(a). In general, the link weight w_{ij} from a_i to a_j is defined by normalizing the coauthorship frequency from a_i as

$$w_{ij} = \frac{f_{ij}}{\sum_{k=1}^{n} f_{ik}}. \tag{6.12}$$

This normalization ensures that the weights of an author's relationships sum to one, as shown in Figure 6.4(b) for C_1.

For each community, a weighted coauthorship graph can be easily built. Intuitively, the generated coauthorship weights express valuable information



(a) (b)

Figure 6.4: Coauthorship graph with: (a) coauthorship frequency, and (b) normalized weight.

which should, and can, be taken into account for discovering the authorities of the authors within the community. We therefore utilize AuthorRank [83], a modification of PageRank [18], to measure the authority for the authors within this community as

$$p(a_i|C_k) = (1-\alpha)\frac{1}{N_a(C_k)} + \alpha \sum_{j=1}^{N_a(C_k)} w_{ij} \cdot p(a_j|C_k), \tag{6.13}$$

where $N_a(C_k)$ is the number of authors in the community C_k, and $p(a_i|C_k)$ is the authority (i.e., AuthorRank) of the author a_i satisfying $\sum_i p(a_i|C_k) = 1$. The AuthorRank can be calculated with the same iterative algorithm used by PageRank.

6.4.2 Community-Sensitive AuthorRank

The AuthorRank described above calculates the authorities for the authors within a community, but it is independent of any particular query topic. To identify a set of experts for a given query, we propose a community-sensitive AuthorRank to generate query-specific authority scores for authors at query time.

We precompute the authority scores offline for each community, as with ordinary AuthorRank. At query time, these authority scores are combined based on the communities of the query to form a composite AuthorRank score for those associated authors. Given a query q, we compute the probability for each community C_k the following:

$$p(C_k|q) = \frac{p(C_k) \cdot p(q|C_k)}{p(q)} \propto p(C_k) \prod_{t_i \in q} p(t_i|C_k), \tag{6.14}$$

where $p(t_i|C_k)$ is easily computed from the community language model as Eq. (6.7). The quantity $p(C_k)$ is not as straightforward. We model it as related to the number of authors $N_a(C_k)$ and the average citation per paper $N_c(C_k)$ in the community c_k; that is

$$p(C_k) \propto N_a(C_k) \cdot \log(10 + N_c(C_k)). \tag{6.15}$$

The number of authors reflects the size of the community, and the average citation per paper reflects the quality of the community. Therefore, the underlying idea is that the community prior is proportional to the size and quality of the community.

According to Eq. (6.14), we retrieve top-k communities that are highly related to the query. Finally, we compute the query-sensitive authority score for each author as follows,

$$p(a_i|q) = \sum_k p(C_k|q)p(a_i|C_k). \tag{6.16}$$

The authors are ranked according to this composite score $p(a_i|q)$. The above community-sensitive AuthorRank has the following probabilistic interpretation. Note that Eq. (6.16) can be reformulated as

$$p(a_i|q) \propto \sum_k p(C_k)p(q|C_k)p(a_i|C_k). \tag{6.17}$$

Suppose C_k be a "virtual" document, it becomes the document based model as Eq. (6.2). Thus the community-sensitive AuthorRank can be regarded as

a high-level document-based model that captures the high-level and general aspects for a given query.

6.4.3 Ranking Refinement Strategy

Based on the document-based model and the community-sensitive AuthorRank (i.e., community-based model), we obtain two kinds of ranking results $\vec{Rd}$ and $\vec{Rc}$, which reflect the authors' expertise from different perspectives. The ranking list $\vec{Rd}$ captures more specific and detailed aspects matching with the given query, as it measures the contribution of each document individually. In contrast, the ranking list $\vec{Rc}$ reflects more general and abstract aspects matching with the given query. In other words, if the document-based model is good for capturing the low-level and specific queries, then the community-sensitive AuthorRank should be good for capturing the high-level and general queries. Therefore, we consider the ranking refinement strategy by leveraging the community-sensitive AuthorRank to boost the document-based model.

In order to measure the similarity and diversity between two ranking results, we utilize a measurement, similar to the Jaccard coefficient, which is defined as the size of the intersection divided by the size of the union of these two top-k ranking results,

$$J = \frac{\left|\vec{Rd} \bigcap \vec{Rc}\right|}{\left|\vec{Rd} \bigcup \vec{Rc}\right|}. \tag{6.18}$$

This measurement implies the following meanings: a large value is reached if the community-sensitive AuthorRank could retrieve many common authors within the top-k results as identified by the document-based model. In this case, the community-sensitive AuthorRank may contribute a lot to refine the document-based model; otherwise vice versa. Based on this scheme, we adopt this measurement for an adaptive ranking refinement as follows. Let $Rd(a_i)$

be the rank of author a_i in $\vec{Rd}$. Suppose $\hat{Rc}$ be the subset of $\vec{Rc}$ consisting of the intersected authors ($\vec{Rd} \bigcap \vec{Rc}$), and let $\hat{Rc}(a_i)$ be the rank of author a_i in $\hat{Rc}$. For each author a_i in $\vec{Rd}$, we define a refined score $S(a_i)$ based on the following function

$$S(a_i) = \frac{1}{Rd(a_i)} + \delta(a_i) \cdot J \cdot \frac{1}{\hat{Rc}(a_i)}, \tag{6.19}$$

where $\delta(a_i) = 1$ if a_i is one of the intersected authors, otherwise $\delta(a_i) = 0$. The intuition behind this method is that the authors, which are identified in both $\vec{Rd}$ and $\vec{Rc}$, should be boosted ahead based on the ranking results $\vec{Rd}$. The new results are ranked according to the refined score $S(a_i)$. By applying the ranking refinement strategy to the previous four different document-based models, we obtain four enhanced models as shown in Table 6.1. The performances of these enhanced models are evaluated and compared in Section 6.5.3.

6.4.4 Overall Algorithm

By unifying the document-based model in Section 6.3 and the enhanced model described above, we summarize the proposed algorithm in Algorithm 2. In the algorithm, note that we first perform preprocessing in a collection, and precompute the following probabilities $p(d_j)$, $p(a_i|d_j)$, $p(C_k)$ and $p(a_i|C_k)$. At query time, our approach is performed as shown in Algorithm 2. Actually, the document-based model is approximately performed using the top-k_1 relevant documents, and meanwhile the community-sensitive AuthorRank is implemented using the top-k_2 relevant communities as well. In Section 6.5.3, we investigate and discuss the effect of these two parameters k_1 and k_2. To deploy the efficient implementations of our scheme, all of the algorithms used in the study are programmed in the C# language. We have implemented the language modeling approach to obtain the initial relevance scores with the Lucene.Net[1] package. For these experiments, the system indexes the collec-

[1] `http://incubator.apache.org/lucene.net/`

tion and does tokenization, stopping and stemming in the usual way.

Algorithm 2 Enhanced Expertise Retrieval Algorithm

Input: Given a query q,

Perform:

1. Retrieve the top-k_1 most relevant documents based on the language model with Eq. (6.6) or Eq. (6.8);
2. Aggregate the expertise $p(a_i, q)$ using the document-based model Eq. (6.2), and then obtain the ranking results $\vec{Rd}$;
3. Identify the top-k_2 most relevant communities according to Eq. (6.14);
4. Compute the community-sensitive AuthorRank with Eq. (6.16), and then obtain the ranking results $\vec{Rc}$;
5. Refine with Eq. (6.19) and get the new ranking results.

Output: Return the ranked experts $\{a_1, a_2, ..., a_k\}$.

6.5 Experimental Evaluation

We evaluate the performance of our proposed models with different settings through an empirical evaluation. In this section, we first introduce the experimental setup, including the dataset and evaluation metrics, and then present the experimental results.

6.5.1 Dataset

The dataset that we study is the DBLP bibliography data, which contains over 1,100,000 XML records as of March 2009. Each record represents an paper that is originally published in conferences, journals, books, etc. One of the XML records is shown in Figure 6.5, and it consists of several elements,

```
<article mdate="2003-11-24" key="journals/cj/Fuhr92">
    <author>Norbert Fuhr</author>
    <title>Probabilistic Models in Information Retrieval.</title>
    <pages>243-255</pages>
    <year>1992</year>
    <volume>35</volume>
    <journal>Comput. J.</journal>
    <number>3</number>
    <url>db/journals/cj/cj35.html#Fuhr92</url>
</article>
```

Figure 6.5: An example of the DBLP XML records.

such as "author", "title", "journal/conference", etc. In total, we gather about 700,000 author names from DBLP XML records, each of which can be an expert candidate. As the DBLP records are limited to represent the papers, we conduct a similar method employed in [37] to extend the information using Google Scholar. For each paper, we use the title as the query to search in Google Scholar and select the top 10 returned records as the supplemental data for this paper. The metadata (HTML pages) crawled from Google Scholar is up to 30GB. This process is done automatically by a crawler and a parser, and the citation of the paper in Google Scholar is obtained at the same time. In addition, we collect the community information according to the journals and conferences, and the total number of valid communities is 3,143. For each community, we regard all the paper titles as the community context, and construct the community coauthorship network for the affiliated authors. In summary, the data collection for experiments include 1,184,678 papers, 696,739 authors, and 3,143 communities. Table 6.2 gives the statistics of the DBLP collection for experiments.

6.5.2 Assessments and Evaluation Metrics

In order to measure the performance of our proposed methods, we manually created the ground truth because of the scarcity of such data that can be

Table 6.2: Statistics of the DBLP collection.

Property	**#of entities**
Number of papers	1,184,678
Number of authors	696,739
Number of communities	3,143

examined publicly. For each query, a list of experts is collected through the method of pooled relevance judgments with human assessment efforts. As shown in Table 6.3, the benchmark dataset used for the evaluation contains 17 query topics and 17 expert lists.

For the evaluation of the task, three different metrics are employed to measure the performance of our proposed models, including precision at rank n (P@n), mean average precision (MAP), bpref [19]. P@n measures the fraction of the top-n retrieved results that are relevant experts for the given query, which is defined as

$$P@n = \frac{\#\text{ relevant experts in top } n \text{ results}}{n}.$$

R-precision (R-prec) is defined as the precision at rank R where R is the number of relevant candidates for the given query. Average precision (AP) emphasizes returning more relevant documents earlier. For a single query, AP is defined as the average of the P@n values for all relevant documents:

$$AP = \frac{\sum_{n=1}^{N}(P@n * \text{rel}(n))}{R},$$

where n is the rank, N the number retrieved, and rel(n) is a binary function indicating the relevance of a given rank. MAP is the mean value of the average precisions computed for all the queries. Beside the measurement of precisions, Bpref [19] is a good score function that evaluates the performance from a

Table 6.3: Benchmark dataset of 17 queries.

Topic	**#Expert**
Boosting	56
Information Extraction	20
Intelligent Agents	29
Machine Learning	42
Natural Language Processing	43
Planning	34
Semantic Web	45
Support Vector Machine	31
Ontology Alignment	55
Probabilistic Relevance Model	13
Information Retrieval	23
Language Model For Information Retrieval	12
Face Recognition	21
Semi Supervised Learning	21
Reinforcement Learning	17
Privacy Preservation	17
Kernel Methods	22

different view, i.e., the number of non-relevant candidates. It is formulated as

$$\text{bpref} = \frac{1}{R}\sum_{r=1}^{N}(1 - \frac{\#n \text{ ranked higher than } r}{R}),$$

where r is a relevant candidate and n is a member of the first R candidates judged non-relevant as retrieved by the system. In our experiments, we report the results of P@10, P@20, P@30, R-prec, MAP, and bpref.

6.5.3 Experimental Results

The presentation of the experiments is organized in the following three aspects. First the experiments are performed to compare the document-based models with different settings. Then we examine the performance of the enhanced models after the ranking refinement. Finally, we discuss the effect of two parameters by the empirical studies, and show some detailed and intermediate results.

Comparison of Document-based Models

To validate the effect of the community-based smoothing method, we evaluate and compare the performance of four document-based methods, including the basedline $DM(b)$, weighted model $DM(w)$, and their smoothed models $DM(bc)$ and $DM(wc)$. The results of these four methods are shown in Table 6.4. The first part shows the absolute precisions of these methods, and the second part illustrates the percentages of relevant improvements.

According to the first part, it is obvious that $DM(wc)$ achieves the best performance among the document-based models in all the metrics, such as 0.5265 for P@20 and 0.3771 for MAP. When looking at the relative improvements, we can see that $DM(bc)$ improves over $DM(b)$ in all metrics, such as 4.4% for P@10 and 4.09% for MAP. Similarly, $DM(wc)$ improves over $DM(w)$

from 1.85% to 4.68% in most metrics besides P@10 (it is harder to be improved as $DM(w)$ has been improved a lot over $DM(b)$). This is because the smoothing method using community context can boost the performance of the language model so as to improve the document-based model for expertise retrieval. The comparisons of $DM(w)/DM(b)$ and $DM(wc)/DM(b)$ show that $DM(w)$ and $DM(wc)$ greatly improve the baseline $DM(b)$, which confirms the importance to consider the document prior in the document-based model. The above experimental results demonstrate the effectiveness of the community-based smoothing method.

Comparison of Enhanced Models

In this subsection, we consider the question whether our proposed enhanced method can boost the performance by incorporating the document-based model with the community-sensitive AuthorRank. In Table 6.5, we present the results of four enhanced models. A quick scan of the table reveals that $EDM(wc)$ always outperforms other methods for all the metrics. In this table, we can see, as expected, that our proposed enhanced models perform better than their corresponding document-based models.

As for the MAP metric, we measure a precision of 0.4089 for $EDM(wc)$, which improves $DM(wc)$ by 8.43%. Similar results are shown in the comparisons of $EDM(b)/DM(b)$, $EDM(bc)/DM(bc)$ and $EDM(w)/DM(w)$, and their relative improvements are 11.44%, 10.85% and 10.92% for MAP, respectively. In terms of the comparisons using other metrics, we observe similar substantial improvements in the enhanced models. By comparing the precisions P@10, P@20 and P@30, an interesting observation is seen that the quantities of improvements in P@20 and P@30 are more significant than those in P@10. All the experimental results demonstrate the effectiveness of the enhanced model, which could further boost the performance of document-

Table 6.4: Comparison of different document-based methods. The percentages of relative improvements are shown in the lower part.

Method	P@10	P@20	P@30	R-prec	MAP	bpref
DM(b)	0.5353	0.45	0.3726	0.4316	0.2897	0.3524
DM(bc)	0.5588	0.4647	0.3824	0.4417	0.3015	0.3621
DM(w)	0.6882	0.5029	0.4235	0.4845	0.3633	0.4159
DM(wc)	**0.6882**	**0.5265**	**0.4314**	**0.4943**	**0.3771**	**0.4279**
DM(bc)/DM(b)	+4.40%	+3.27%	+2.63%	+2.34%	+4.09%	+2.78%
DM(wc)/DM(w)	0%	+4.68%	+1.85%	+2.03%	+3.79%	+2.89%
DM(w)/DM(b)	+28.57%	+11.76%	+13.68%	+12.26%	+25.43%	+18.02%
DM(wc)/DM(b)	+28.57%	+16.99%	+15.79%	+14.53%	+30.19%	+21.44%

based models. Moreover, the improvements made in the enhanced model are consistent and promising. Therefore, it is very essential and promising to consider the enhanced models for expertise retrieval.

Discussion and Detailed Results

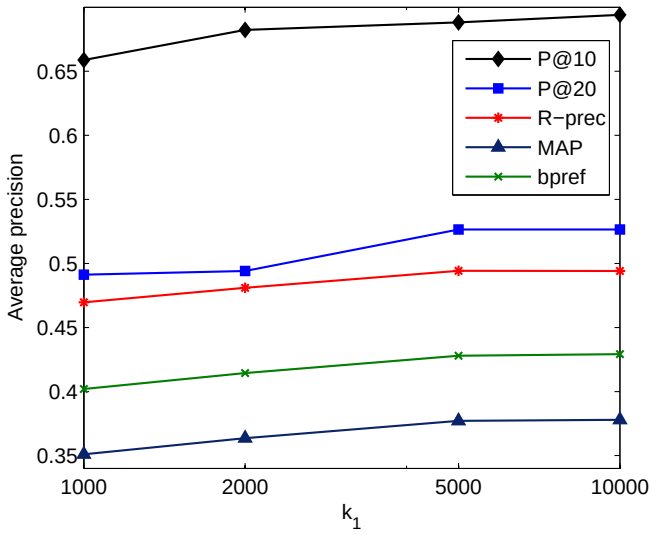
We have shown the effectiveness and improvement of our proposed document-based models and enhanced models. The parameters k_1 and k_2 used in previous subsections are set to 5,000 and 10, individually. As mentioned before, we only retrieve the top-k_1 relevant documents for the document-based model, and identify top-k_2 relevant communities for the community-sensitive AuthorRank as well. To investigate the effect of these two parameters, we designed the following experiments.

To examine the effect of k_1, we choose the best document-based model $DM(wc)$, and evaluate it with 4 different values (from 1,000 to 10,000). The experimental results for different k_1 are illustrated in Figure 6.6(a). In this figure, we can see the performance becomes better for greater k_1 used in the document-based model. We believe the reason is that more documents can better capture the complete expertise. However, larger k_1 may result in longer processing time. Therefore, a good tradeoff is to set $k_1 = 5000$. To investigate the effect of k_2, we fix $k_1 = 5000$, and choose to compare the model $EDM(wc)$ with several different values from 0 to 50. Here, $k_2 = 0$ in $EDM(wc)$ represents its document-based model $DM(wc)$. As shown in Figure 6.6(b), when incorporating the community-sensitive AuthorRank in the enhanced model ($k_2 > 0$), the performance is improved compared to the document-based model ($k_2 = 0$). The precisions first increase then level off as k_2 grows. In general, the enhanced model $EDM(wc)$ is relatively robust for different k_2, and achieves good results when $k_2 = 10$.

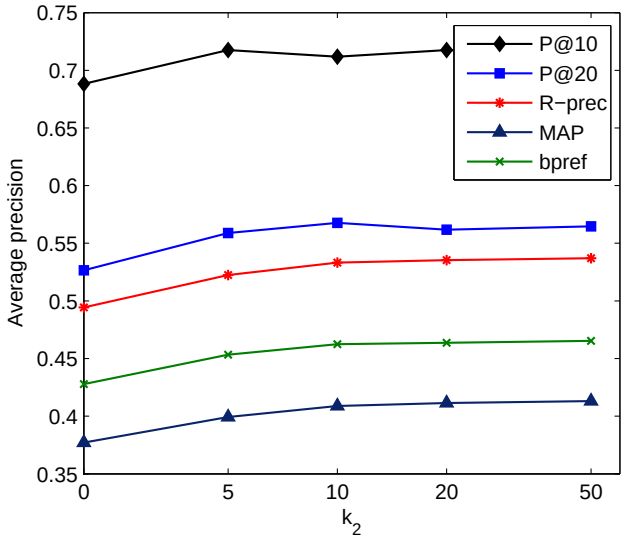
To gain a better insight into the proposed enhanced model, we choose

Table 6.5: Comparison of different enhanced methods. The percentages of relative improvements are shown in the lower part.

Method	P@10	P@20	P@30	R-prec	MAP	bpref
EDM(b)	0.5882	0.4971	0.4196	0.4716	0.3228	0.38933
EDM(bc)	0.5941	0.5059	0.4275	0.4803	0.3342	0.39879
EDM(w)	0.7059	0.55	0.4608	0.5317	0.403	0.45839
EDM(wc)	**0.7118**	**0.5677**	**0.4628**	**0.5332**	**0.4089**	**0.46241**
EDM(b)/DM(b)	+9.89%	+10.46%	+12.63%	+9.28%	+11.44%	+10.49%
EDM(bc)/DM(bc)	+6.31%	+8.86%	+11.79%	+8.75%	+10.85%	+10.12%
EDM(w)/DM(w)	+2.56%	+9.36%	+8.79%	+9.75%	+10.92%	+10.22%
EDM(wc)/DM(wc)	+3.42%	+7.82%	+7.27%	+7.86%	+8.43%	+8.06%



(a) $DM(wc)$



(b) $EDM(wc)$

Figure 6.6: The effect of varying the parameters (k_1 and k_2) in (a) the document-based model $DM(wc)$ and (b) the enhanced model $EDM(wc)$.

Table 6.6: The detailed results of the community-sensitive AuthorRank for the query "machine learning." The first row is the top-5 communities for the query, and the rest part lists the top-10 author lists ranked by their authorities in the community.

journals/ML	conf/ICML	conf/NIPS	journals/JMLR	conf/ECML
Pat Langley	Andrew W. Moore	Terrence J. Sejnowski	Michael I. Jordan	Saso Dzeroski
Robert E. Schapire	Sridhar Mahadevan	Michael I. Jordan	Yoram Singer	Johannes Frnkranz
Manfred K. Warmuth	Thomas G. Dietterich	Geoffrey E. Hinton	Tong Zhang	Gerhard Widmer
Thomas G. Dietterich	Prasad Tadepalli	Peter Dayan	Francis R. Bach	Ivan Bratko
Yoram Singer	Michael L. Littman	Christof Koch	Olivier Bousquet	Enric Plaza
Ryszard S. Michalski	Pat Langley	Klaus-Robert Mller	Klaus-Robert Mller	Pavel Brazdil
Michael J. Pazzani	Andrew McCallum	Zoubin Ghahramani	Bernhard Schlkopf	Birgit Tausend
Dana Angluin	Thorsten Joachims	Michael Mozer	Andr Elisseeff	Stephen Muggleton
Avrim Blum	Satinder P. Singh	Bernhard Schlkopf	Koby Crammer	Floriana Esposito
Leo Breiman	Michael I. Jordan	Satinder P. Singh	Ingo Steinwart	Stan Matwin

Table 6.7: The top-10 expert lists retrieved by the document-based model $DM(wc)$, the community-sensitive AuthorRank, and the enhanced model $EDM(wc)$, for the query "machine learning."

DM(wc)	Authorities	EDM(wc)
Pat Langley	Pat Langley	Pat Langley
Thomas G. Dietterich	Robert E. Schapire	Thomas G. Dietterich
Sumio Watanabe	Manfred K. Warmuth	Sumio Watanabe
David E. Goldberg	Yoram Singer	David E. Goldberg
Tom M. Mitchell	Thomas G. Dietterich	Avrim Blum
Avrim Blum	Michael I. Jordan	Tom M. Mitchell
Ivan Bratko	Satinder P. Singh	Sanjay Jain
Donald Michie	Sanjay Jain	Ivan Bratko
Carl H. Smith	John Shawe-Taylor	Donald Michie
J. Ross Quinlan	Michael J. Pazzani	Michael I. Jordan

the query "machine learning" as the case to detail the combination of the community-sensitive AuthorRank and the document-based model, and to show the intermediate results as well. We first present the detailed results of the community-sensitive AuthorRank in Table 6.6. According to Eq. (6.14), the top-5 relevant communities to the query "machine learning" are identified in the first row of Table 6.6, which are the "Machine Leaning journal", "ICML conference", "NIPS conference", "JMLR journal", and "ECML conference." Using the AuthorRank, we could easily obtain their authorities for these communities. The top-10 author lists ranked by their authorities are listed in Table 6.6. As we can see, the proposed method can capture the right communities as well as the authoritative authors, such as "Andrew W. Moore" in ICML and "Michael I. Jordan" in NIPS. With the top-k identified communities, the community-sensitive AuthorRank is employed to generate the query-sensitive authorities. In this case, the top-10 author list ranked by the query-sensitive authorities is shown in the second column of Table 6.7. The other two columns in Table 6.7, reports the top-10 expert lists retrieved by $DM(wc)$ and $EDM(wc)$, respectively. We observe that a slight change occurs in the output of $EDM(wc)$ in contrast to that of $DM(wc)$, which would boost the persons retrieved by both the document-based model and the community-sensitive AuthorRank.

6.6 Summary

In this chapter we present the community-aware strategies for enhancing expertise retrieval, including the new smoothing method with the community context and the community-sensitive AuthorRank based on the coauthorship networks, which are motivated by the observation that the community provides valuable and distinctive information along with the documents and the experts. We not only formally define and quantify these two strategies, but

also propose the adaptive ranking refinement method to incorporate both ranking results for an effective enhanced model. We apply the proposed models to the expert finding task on the DBLP bibliography data. Extensive experiments show that the improvements of our enhanced models are significant and consistent.

□ **End of chapter.**

Chapter 7

Conclusions

In this chapter, we summarize the key research results presented in this thesis, and discuss some possible future research work.

7.1 Summary

This thesis aims to develop a general framework to make use of the content and graph information effectively by leveraging information retrieval, machine learning, and knowledge discovery techniques for real-world applications, especially query log analysis and expertise retrieval. To this purpose, we develop scalable automatic content analysis methods and graph-based models to analyze a huge amount of data resources including AOL query logs, online DBLP, Google Scholar, etc. and propose several approaches to tackle various challenging problems. The major achievements and contributions are concluded in the following.

First of all, a novel entropy-biased framework is proposed for modeling bipartite graphs, which intends to find better query representations by diminishing the effect of noisy links and treating heterogeneous query-URL pairs differently for click graphs. The intuition behind this model is common clicks on less frequent but more specific URLs are of greater value than

common clicks on frequent and general URLs. Based on this intuition, the entropy-biased model introduce the *inverse query frequency* to weigh the importance of a click on a certain URL. Moreover, the inverse query frequency is incorporated with raw click frequencies and other information together to achieve better performance. The proposed entropy-biased framework is never explicitly explored or statistically examined for any bipartite graphs in the information retrieval literature.

According to the graph information, there is a lack of constraints to make sure the final relevance of the score propagation on the graph. To tackle this problem, a general Co-HITS algorithm is developed to incorporate the bipartite graph with the content information from both sides as well as the constraints of relevance. Moreover, the algorithm is investigated based on two frameworks, including the iterative and the regularization frameworks. For the iterative framework, it contains HITS and personalized PageRank as special cases. In the regularization framework, we successfully build a connection with HITS, and develop a new cost function to consider the direct relationship between two entity sets, which leads to a significant improvement over the baseline method.

In contrast to the traditional document retrieval, expertise retrieval is a high-level information retrieval with more heterogeneous information environment. The objective of this thesis is to propose a general Web mining framework to combine the content with the graph information as well as other kinds of information effectively. Therefore, a new expert finding framework is proposed based on the large-scale DBLP bibliography and its supplemental data from Google Scholar. In addition, a weighted language model is employed to aggregate the expertise of a candidate from the associated documents. The model not only considers the relevance of documents against a given query, but also incorporates the importance of the documents in the

form of document priors. Moreover, a graph-based regularization method is integrated to enhance the model by refining the relevance scores of the documents with respect to the query.

Previous algorithms mainly consider the documents associated with the experts, while ignoring the community information that is affiliated with the documents and the experts. Motivated by the observation that communities could provide valuable insight and distinctive information, we develop two community-aware strategies to enhance the expertise retrieval. We first propose a new smoothing method using the community context for statistical language model, which is employed to identify the most relevant documents so as to reflect the expertise retrieval in the document-based model. Furthermore, we propose a query-sensitive AuthorRank to model the authors' authorities based on the community coauthorship networks, and develop an adaptive ranking refinement method to enhance the expertise retrieval.

7.2 Future Work

Although a substantial number of promising achievements on Web mining and its applications have been presented in this thesis, there are still numerous open issues that need to be further explored in future work.

First, the ultimate goal of query log analysis is to understand what users want and present to them. As the click graph is an important technique for describing the information provided in the query log, one natural extension is to combine the user's individual click graph and session information for personalized query log analysis. In addition, it is reasonable to incorporate the click graph with other information, such as query-flow model and user modeling. Although we have developed the efficient algorithms for modeling and analyzing click graphs, there is still a huge challenge to understand a user's query intent. Effective query understanding is critical to a successful

search and navigation application. Therefore, when conducting the search algorithms, it is important to consider how users will enter their queries and how they can easily find the needed information. Traditional approaches have been applied to single intent level, for example, "job intent" or "product intent", which can be regarded as a step toward this goal. To identify user's query intent in general cases, it is necessary to develop more powerful models to achieve this goal.

Second, it would be interesting to apply the generalized Co-HITS algorithm to the expertise retrieval task, since the author-paper bipartite graph with content information can be obtained from the expertise retrieval data. In order to further improve the performance of the expertise retrieval methods, some other information related to the researcher people should be utilized and incorporated into a unified learning process, such as the profile and social information of the expert candidates (researchers). The challenging problems include how to find and extract the profile as well as the social information, and how to integrate different information together.

Third, expertise retrieval currently is limited to a particular domain or intranet, and a much more challenging task would be to perform expertise retrieval on the Web. One important task is to identify relevant experts or trusted people who can offer solutions in a timely and human manner. Furthermore, approaches to create a global expert and friend recommendation social network should be further studied to not only facilitate Web-scale expert and social search but also leverage the results to rate online contents.

Another important issue is to develop advanced methodology for identifying and scoring the relevant documents which match the close meaning, not the exact terms, for the given query. This challenge may be solved if it is effective to build an automatic matching system between similar words and concepts with natural language processing and machine translation tech-

niques. In addition to the combination of content and graph information, more study is needed for our framework to incorporate with multiple other sources.

Last, but not least, we may extend our framework by exploring other machine learning techniques, such as learning to rank. Also we will apply our methodologies and algorithms to solve a variety of applications in data mining and information retrieval, such as entity retrieval, personalized search, and online social media search. New search tasks and interfaces for the presentation of search results, like literature retrieval, expert search, and query suggestion, come with the need to rank entities, such as persons, organizations and query, instead of documents or text passages.

□ **End of chapter.**

Appendix A

List of Publications

1. **Hongbo Deng**, Irwin King, Michael R. Lyu. Enhancing Expertise Retrieval Using Community-aware Strategies. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management* (Hong Kong, China, Nov. 2-6, 2009). CIKM 2009.

2. **Hongbo Deng**, Irwin King, Michael R. Lyu. Entropy-biased Models for Query Representation on the Click Graph. In *Proceedings of the 32nd Annual ACM SIGIR Conference on Research and Development in Information Retrieval* (Boston, MA, USA, July 19-23, 2009). SIGIR 2009. Pages: 339-346. (Acceptance rate: 78/494 = 16%)

3. **Hongbo Deng**, Michael R. Lyu and Irwin King. A Generalized Co-HITS Algorithm and Its Application to Bipartite Graphs. In *Proceedings of the 15th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (Paris, France, June 28th-July 1st, 2009). KDD 2009. Pages: 239-248. (Acceptance rate: 105/561 = 19%)

4. **Hongbo Deng**, Michael R. Lyu and Irwin King. Effective Latent Space Graph-based Re-ranking Model with Global Consistency. In *Proceedings of the 2nd ACM International Conference on Web Search and Data*

Mining (Barcelona, Spain, Feb. 9-12, 2009). WSDM 2009. Pages: 212-221. (Acceptance rate: 29/170 = 17%)

5. **Hongbo Deng**, Irwin King and Michael R. Lyu. Formal Models for Expert Finding on DBLP Bibliography Data. In *Proceedings of the 8th IEEE International Conference on Data Mining* (Pisa, Italy, Dec. 15-19, 2008). ICDM 2008. Pages: 163-172. (Acceptance rate: 70/724 = 10%)

6. **Hongbo Deng**, Jianke Zhu, Michael R. Lyu and Irwin King. Two-Stage Multi-Class AdaBoost for Facial Expression Recognition. In *Proceedings of International Joint Conference on Neural Networks* (Florida, USA, Aug.12-17, 2007). IJCNN 2007. Pages: 3005-3010.

Under Review:

1. **Hongbo Deng**, Irwin King, Michael R. Lyu. A Weighted Language Model for Expert Search with Graph-based Regularization. Submitted to TKDE journal.

□ **End of chapter.**

Bibliography

[1] Dblp bibliography. URL: `http://www.informatik.uni-trier.de/~ley/db/`, 2007.

[2] Google scholar. URL: `http://scholar.google.com/`, 2008.

[3] A. Agarwal, S. Chakrabarti, and S. Aggarwal. Learning to rank networked entities. In *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 14–23, 2006.

[4] J. Alpert and N. Hajaj. We knew the web was big... *The Official Google Blog*, July 25, 2008.

[5] R. Baeza-Yates, B. Ribeiro-Neto, et al. *Modern information retrieval*. Addison-Wesley Harlow, England, 1999.

[6] R. A. Baeza-Yates, C. A. Hurtado, and M. Mendoza. Query recommendation using query logs in search engines. In *EDBT Workshops*, pages 588–596, 2004.

[7] R. A. Baeza-Yates and A. Tiberi. Extracting semantic relations from query logs. In *KDD*, pages 76–85, 2007.

[8] K. Balog, L. Azzopardi, and M. de Rijke. Formal models for expert finding in enterprise corpora. In *Proceedings of the 29th International*

ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), pages 43–50, 2006.

[9] K. Balog, T. Bogers, L. Azzopardi, M. de Rijke, and A. van den Bosch. Broad expertise retrieval in sparse data environments. In *Proceedings of the 30th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 551–558, 2007.

[10] K. Balog and M. de Rijke. Determining expert profiles (with an application to expert finding). In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2657–2662, 2007.

[11] D. Beeferman and A. L. Berger. Agglomerative clustering of a search engine query log. In *KDD*, pages 407–416, 2000.

[12] S. M. Beitzel, E. C. Jensen, D. D. Lewis, A. Chowdhury, and O. Frieder. Automatic classification of web queries using very large unlabeled query logs. *ACM Trans. Inf. Syst.*, 25(2), 2007.

[13] M. Belkin, I. Matveeva, and P. Niyogi. Regularization and semi-supervised learning on large graphs. In *COLT*, pages 624–638, 2004.

[14] M. Belkin and P. Niyogi. Semi-supervised learning on riemannian manifolds. *Machine Learning*, 56(1-3):209–239, 2004.

[15] M. Bilenko and R. W. White. Mining the search trails of surfing crowds: identifying relevant websites from user activity. In *WWW*, pages 51–60, 2008.

[16] A. Blum and T. M. Mitchell. Combining labeled and unlabeled sata with co-training. In *COLT*, pages 92–100, 1998.

[17] A. Borodin, G. O. Roberts, J. S. Rosenthal, and P. Tsaparas. Link analysis ranking: algorithms, theory, and experiments. *ACM Trans. Internet Techn.*, 5(1):231–297, 2005.

[18] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, 30(1-7):107–117, 1998.

[19] C. Buckley and E. M. Voorhees. Retrieval evaluation with incomplete information. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 25–32, 2004.

[20] C. J. C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. N. Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 89–96, 2005.

[21] Y. Cao, J. Liu, S. Bao, and H. Li. Research on expert search at enterprise track of trec 2005. In *Proceedings of TREC 2005*, 2005.

[22] P. Carrington, J. Scott, and S. Wasserman. *Models and methods in social network analysis*. Cambridge University Press, 2005.

[23] S. Chakrabarti. Data mining for hypertext: A tutorial survey. *SIGKDD Explorations*, 1(2):1–11, 2000.

[24] S. Chakrabarti, B. Dom, R. Kumar, P. Raghavan, S. Rajagopalan, A. Tomkins, D. Gibson, and J. M. Kleinberg. Mining the web's link structure. *IEEE Computer*, 32(8):60–67, 1999.

[25] L.-W. Chan. Analysis of the internal representations in neural networks for machine intelligence. In *AAAI*, pages 578–583, 1991.

[26] O. Chapelle, B. Schölkopf, and A. Zien. *Semi-supervised learning*. MIT press, 2006.

[27] D. Cohn and H. Chang. Learning to probabilistically identify authoritative documents. In *Proceedings of the 17th International Conference on Machine Learning*, pages 167–174, 2000.

[28] D. A. Cohn and T. Hofmann. The missing link - a probabilistic model of document content and hypertext connectivity. In *Advances in Neural Information Processing Systems*, pages 430–436, 2000.

[29] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[30] N. Craswell, I. Soboroff, and A. de Vries. Overview of the trec-2005 enterprise track. In *Proceedings of TREC 2005*.

[31] N. Craswell and M. Szummer. Random walks on the click graph. In *SIGIR*, pages 239–246, 2007.

[32] N. Craswell, O. Zoeter, M. Taylor, and B. Ramsey. An experimental comparison of click position-bias models. In *WSDM*, pages 87–94, 2008.

[33] H. Cui, J.-R. Wen, J.-Y. Nie, and W.-Y. Ma. Probabilistic query expansion using query logs. In *WWW*, pages 325–332, 2002.

[34] T. Davis. *Direct Methods for Sparse Linear Systems*. Society for Industrial Mathematics, 2006.

[35] A. P. de Vries and T. Rölleke. Relevance information: a loss of entropy but a gain for idf? In *SIGIR*, pages 282–289, 2005.

[36] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

[37] H. Deng, I. King, and M. R. Lyu. Formal models for expert finding on dblp bibliography data. In *Proceedings of the 8th IEEE International Conference on Data Mining*, pages 163–172, 2008.

[38] H. Deng, I. King, and M. R. Lyu. Enhancing expertise retrieval using community-aware strategies. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM)*, 2009.

[39] H. Deng, I. King, and M. R. Lyu. Entropy-biased models for query representation on the click graph. In *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 339–346, 2009.

[40] H. Deng, M. R. Lyu, and I. King. Effective latent space graph-based re-ranking model with global consistency. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining (WSDM)*, pages 212–221, 2009.

[41] H. Deng, M. R. Lyu, and I. King. A generalized co-hits algorithm and its application to bipartite graphs. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 239–248, 2009.

[42] F. Diaz. Regularizing ad hoc retrieval scores. In *Proceedings of the 2005 ACM CIKM International Conference on Information and Knowledge Management*, pages 672–679, 2005.

[43] C. H. Q. Ding, X. He, P. Husbands, H. Zha, and H. D. Simon. Pagerank, hits and a unified framework for link analysis. In *SIGIR*, pages 353–354, 2002.

[44] Z. Dou, R. Song, and J.-R. Wen. A large-scale evaluation and analysis of personalized search strategies. In *WWW*, pages 581–590, 2007.

[45] G. Dupret and B. Piwowarski. A user browsing model to predict search engine click data from past observations. In *SIGIR*, pages 331–338, 2008.

[46] G. Dupret, B. Piwowarski, C. A. Hurtado, and M. Mendoza. A statistical model of query log generation. In *SPIRE*, pages 217–228, 2006.

[47] O. Etzioni. The world-wide web: Quagmire or gold mine? *Commun. ACM*, 39(11):65–68, 1996.

[48] H. Fang and C. Zhai. Probabilistic models for expert finding. In *Proceedings of the European Conference on Information Retrieval (ECIR)*, pages 418–430, 2007.

[49] N. Fuhr. Probabilistic models in information retrieval. *The Computer Journal*, 35(3):243, 1992.

[50] E. Garfield. *Citation indexing: Its theory and application in science, technology, and humanities*. Wiley New York, 1979.

[51] V. Gudivada, V. Raghavan, W. Grosky, R. Kasanagottu, and D. Markets. Information retrieval on the world wide web. *IEEE Internet Computing*, 1(5):58–68, 1997.

[52] Z. Gyöngyi, H. Garcia-Molina, and J. O. Pedersen. Combating web spam with trustrank. In *VLDB*, pages 576–587, 2004.

[53] T. Haveliwala. Topic-sensitive pagerank: A context-sensitive ranking algorithm for web search. *IEEE Transactions on Knowledge and Data Engineering*, pages 784–796, 2003.

[54] T. Haveliwala, S. Kamvar, and G. Jeh. An analytical comparison of approaches to personalizing PageRank. *Preprint, June*, 2003.

[55] M. Henzinger, R. Motwani, and C. Silverstein. Challenges in web search engines. In *ACM SIGIR Forum*, volume 36, pages 11–22. ACM New York, NY, USA, 2002.

[56] S. Hettich and M. J. Pazzani. Mining for proposal reviewers: lessons learned at the national science foundation. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 862–871, 2006.

[57] T. Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 50–57, 1999.

[58] A. Jain, M. Murty, and P. Flynn. Data clustering: a review. *ACM computing surveys*, 31(3), 1999.

[59] G. Jeh and J. Widom. Scaling personalized web search. In *WWW*, pages 271–279, 2003.

[60] R. Jin, A. G. Hauptmann, and C. Zhai. Title language model for information retrieval. In *Proceedings of the 25th ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 42–48, 2002.

[61] X. Jin, Y. Zhou, and B. Mobasher. Web usage mining based on probabilistic latent semantic analysis. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 197–205. ACM New York, NY, USA, 2004.

[62] T. Joachims. Text categorization with suport vector machines: Learning with many relevant features. In *ECML*, pages 137–142, 1998.

[63] T. Joachims. Transductive inference for text classification using support vector machines. In *ICML*, pages 200–209, 1999.

[64] T. Joachims. Optimizing search engines using clickthrough data. In *KDD*, pages 133–142, 2002.

[65] K. S. Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28(1):11–21, 1972.

[66] R. Jones, B. Rey, O. Madani, and W. Greiner. Generating query substitutions. In *Proceedings of the 15th International Conference on World Wide Web*, pages 387–396, 2006.

[67] S. Kamvar, M. Schlosser, and H. Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. In *Proceedings of the 12th international conference on World Wide Web*, pages 640–651. ACM New York, NY, USA, 2003.

[68] M. Karimzadehgan, R. W. White, and M. Richardson. Enhancing expert finding using organizational hierarchies. In *ECIR*, pages 177–188, 2009.

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

[70] T. Kohonen. An introduction to neural computing. *Neural networks*, 1(1):3–16, 1988.

[71] R. Kosala and H. Blockeel. Web mining research: A survey. *SIGKDD Explorations*, 2(1):1–15, 2000.

[72] O. Kurland and L. Lee. Pagerank without hyperlinks: structural reranking using links induced by language models. In *Proceedings of the 28nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 306–313, 2005.

[73] O. Kurland and L. Lee. Respect my authority!: Hits without hyperlinks, utilizing cluster-based language models. In *Proceedings of the 29nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 83–90, 2006.

[74] J. Lafferty and C. Zhai. Probabilistic relevance models based on document and query generation. *Language modeling for information retrieval*, 13:1–10, 2003.

[75] V. Lavrenko and W. B. Croft. Relevance-based language models. In *SIGIR*, pages 120–127, 2001.

[76] K.-S. Leung, H. Jin, and Z.-B. Xu. An expanding self-organizing neural network for the traveling salesman problem. *Neurocomputing*, 62:267–292, 2004.

[77] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li. Rcv1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397, 2004.

[78] H. Li, Z. Nie, W.-C. Lee, C. L. Giles, and J.-R. Wen. Scalable community discovery on textual data with relations. In *CIKM*, pages 1203–1212, 2008.

[79] J.-Z. Li, J. Tang, J. Zhang, Q. Luo, Y. Liu, and M. Hong. Eos: expertise oriented search using social networks. In *Proceedings of the 16th International Conference on World Wide Web*, pages 1271–1272, 2007.

[80] W. Li, K.-H. Lee, and K.-S. Leung. Generalized regularized least-squares learning with predefined features in a hilbert space. In *NIPS*, pages 881–888, 2006.

[81] X. Li, Y.-Y. Wang, and A. Acero. Learning query intent from regularized click graphs. In *SIGIR*, pages 339–346, 2008.

[82] B. Liu. *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data*. Data-Centric Systems and Applications. Springer, 2007.

[83] X. Liu, J. Bollen, M. L. Nelson, and H. V. de Sompel. Co-authorship networks in the digital library research community. *Inf. Process. Manage.*, 41(6):1462–1480, 2005.

[84] X. Liu and W. B. Croft. Cluster-based retrieval using language models. In *SIGIR*, pages 186–193, 2004.

[85] Y. Liu, B. Gao, T. Liu, Y. Zhang, Z. Ma, S. He, and H. Li. Browserank: Letting web users vote for page importance. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 451–458. ACM New York, NY, USA, 2008.

[86] H. Ma, H. Yang, I. King, and M. R. Lyu. Learning latent semantic relations from clickthrough data for query suggestion. In *CIKM*, pages 709–718, 2008.

[87] C. Macdonald, D. Hannah, and I. Ounis. High quality expertise evidence for expert search. In *ECIR*, pages 283–295, 2008.

[88] C. Macdonald and I. Ounis. Voting for candidates: adapting data fusion techniques for an expert search task. In *Proceedings of the 15th ACM CIKM International Conference on Information and Knowledge Management (CIKM)*, pages 387–396, 2006.

[89] C. Macdonald and I. Ounis. Expertise drift and query expansion in expert search. In *Proceedings of the 16th ACM CIKM International Conference on Information and Knowledge Management (CIKM)*, pages 341–350, 2007.

[90] P. Maes. Agents that reduce work and information overload. *Communication of the ACM*, 37(7):30–40, 1994.

[91] C. Manning, P. Raghavan, and H. Schtze. *Introduction to information retrieval*. Cambridge University Press New York, NY, USA, 2008.

[92] M. Marchiori. The quest for correct information on the web: Hyper search engines. *Computer Networks*, 29(8-13):1225–1236, 1997.

[93] M. T. Maybury, R. J. D'Amore, and D. House. Expert finding for collaborative virtual environments. *Communications of the ACM*, 44(12):55–56, 2001.

[94] A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*, volume 7, 1998.

[95] Q. Mei, D. Cai, D. Zhang, and C. Zhai. Topic modeling with network regularization. In *Proceedings of the 17th International Conference on World Wide Web*, pages 101–110, 2008.

[96] Q. Mei, D. Zhou, and K. W. Church. Query suggestion using hitting time. In *CIKM*, pages 469–478, 2008.

[97] D. M. Mimno and A. McCallum. Expertise modeling for matching papers with reviewers. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 500–509, 2007.

[98] E. Minkov, W. W. Cohen, and A. Y. Ng. Contextual search and name disambiguation in email using graphs. In *Proceedings of the 29th ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 27–34, 2006.

[99] B. Mobasher, R. Cooley, and J. Srivastava. Automatic personalization based on web usage mining. 2000.

[100] M. Newman. Coauthorship networks and patterns of scientific collaboration. *Proceedings of the National Academy of Sciences*, 101(suppl 1):5200–5205, 2004.

[101] Z. Nie, Y. Zhang, J.-R. Wen, and W.-Y. Ma. Object-level ranking: bringing order to web objects. In *Proceedings of the 14th International Conference on World Wide Web*, pages 567–574, 2005.

[102] G. Pass, A. Chowdhury, and C. Torgeson. A picture of search. In *Infoscale*, page 1, 2006.

[103] D. Petkova and W. B. Croft. Hierarchical language models for expert finding in enterprise corpora. In *18th IEEE International Conference on Tools with Artificial Intelligence*, pages 599–608, 2006.

[104] D. Petkova and W. B. Croft. Proximity-based document representation for named entity retrieval. In *Proceedings of the 16th ACM CIKM International Conference on Information and Knowledge Management (CIKM)*, pages 731–740, 2007.

[105] D. Pierrakos, G. Paliouras, C. Papatheodorou, and C. Spyropoulos. Web usage mining as a tool for personalization: A survey. *User Modeling and User-Adapted Interaction*, 13(4):311–372, 2003.

[106] B. Poblete and R. A. Baeza-Yates. Query-sets: using implicit feedback and query patterns to organize web documents. In *WWW*, pages 41–50, 2008.

[107] B. Poblete, C. Castillo, and A. Gionis. Dr. searcher and mr. browser: a unified hyperlink-click graph. In *CIKM*, pages 1123–1132, 2008.

[108] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *Proceedings of the 21th ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 275–281, 1998.

[109] T. Qin, T.-Y. Liu, X.-D. Zhang, D.-S. Wang, W.-Y. Xiong, and H. Li. Learning to rank relational objects and its application to web search. In *Proceedings of the 17th International Conference on World Wide Web*, pages 407–416, 2008.

[110] F. Radlinski and T. Joachims. Query chains: learning to rank from implicit feedback. In *KDD*, pages 239–248, 2005.

[111] F. Radlinski and T. Joachims. Active exploration for learning rankings from clickthrough data. In *KDD*, pages 570–579, 2007.

[112] S. Robertson. The probability ranking principle in IR. *Journal of documentation*, 33(4):294–304, 1977.

[113] S. Robertson. Understanding inverse document frequency: on theoretical arguments for IDF. *Journal of Documentation*, 60:503–520, 2004.

[114] S. Robertson and K. Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27(3), 1976.

[115] S. Robertson, C. Van Rijsbergen, and M. Porter. Probabilistic models of indexing and searching. In *Proceedings of the 3rd annual ACM conference on Research and development in information retrieval*, pages 35–56. Butterworth & Co. Kent, UK, UK, 1980.

[116] M. A. Rodriguez and J. Bollen. An algorithm to determine peer-reviewers. *CoRR*, abs/cs/0605112, 2006.

[117] T. Roelleke and J. Wang. Tf-idf uncovered: a study of theories and probabilities. In *SIGIR*, pages 435–442, 2008.

[118] C. Rosenberg, M. Hebert, and H. Schneiderman. Semi-supervised self-training of object detection models. In *WACV/MOTION*, pages 29–36, 2005.

[119] R. Rubin. *Foundations of library and information science. Second ed.* Neal-Schuman Publishers., 2004.

[120] G. Salton. The SMART retrieval systemexperiments in automatic document processing. 1971.

[121] G. Salton and C. Buckley. Term-Weighting Approaches in Automatic Text Retrieval. *Information Processing and Management*, 24(5):513–23, 1988.

[122] G. Salton, E. Fox, and H. Wu. Extended Boolean information retrieval. 1983.

[123] G. Salton, A. Wong, and C. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.

[124] F. Sebastiani. Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1):1–47, 2002.

[125] P. Serdyukov, H. Rode, and D. Hiemstra. Modeling multi-step relevance propagation for expert finding. In *CIKM*, pages 1133–1142, 2008.

[126] C. E. Shannon. Prediction and entropy of printed english. *The Bell System Technical Journal*, 30:50–64, 1950.

[127] D. Shen, R. Pan, J.-T. Sun, J. J. Pan, K. Wu, J. Yin, and Q. Yang. Query enrichment for web-query classification. *ACM Trans. Inf. Syst.*, 24(3):320–352, 2006.

[128] D. Shen, M. Qin, W. Chen, Q. Yang, and Z. Chen. Mining web query hierarchies from clickthrough data. In *AAAI*, pages 341–346, 2007.

[129] D. Shen, J.-T. Sun, Q. Yang, and Z. Chen. Building bridges for web query classification. In *SIGIR*, pages 131–138, 2006.

[130] V. Sindhwani and S. S. Keerthi. Large scale semi-supervised linear svms. In *SIGIR*, pages 477–484, 2006.

[131] A. Singhal, G. Salton, M. Mitra, and C. Buckley. Document length normalization. *Inf. Process. Manage.*, 32(5):619–633, 1996.

[132] A. Smola and R. Kondor. Kernels and regularization on graphs. *Conference on Learning Theory, COLT/KW*, 2003.

[133] I. Soboroff, A. de Vries, and N. Craswell. Overview of the trec-2006 enterprise track. In *Proceedings of TREC 2006*.

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

[135] J. Srivastava, R. Cooley, M. Deshpande, and P. Tan. Web usage mining: Discovery and applications of usage patterns from web data. *SIGKDD explorations*, 1(2):12–23, 2000.

[136] J. Teevan, S. T. Dumais, and D. J. Liebling. To personalize or not to personalize: modeling queries with variation in user intent. In *SIGIR*, pages 163–170, 2008.

[137] X. Wang and C. Zhai. Learn from web search logs to organize search results. In *SIGIR*, pages 87–94, 2007.

[138] X. Wei and W. B. Croft. Lda-based document models for ad-hoc retrieval. In *Proceedings of the 29th Annual International ACM SIGIR*

Conference on Research and Development in Information Retrieval, pages 178–185, 2006.

[139] J.-R. Wen, J.-Y. Nie, and H. Zhang. Clustering user queries of a search engine. In *WWW*, pages 162–168, 2001.

[140] R. W. White and S. M. Drucker. Investigating behavioral variability in web search. In *WWW*, pages 21–30, 2007.

[141] S. Wong, W. Ziarko, and P. Wong. Generalized vector spaces model in information retrieval. In *Proceedings of the 8th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 18–25. ACM New York, NY, USA, 1985.

[142] G.-R. Xue, H.-J. Zeng, Z. Chen, Y. Yu, W.-Y. Ma, W. Xi, and W. Fan. Optimizing web search using web click-through data. In *CIKM*, pages 118–126, 2004.

[143] Y. Yang and X. Liu. A re-examination of text categorization methods. In *SIGIR*, pages 42–49, 1999.

[144] H. Zaragoza, H. Rode, P. Mika, J. Atserias, M. Ciaramita, and G. Attardi. Ranking very many typed entities on wikipedia. In *CIKM*, pages 1015–1018, 2007.

[145] C. Zhai. Statistical language models for information retrieval: A critical review. *Foundations and Trends in Information Retrieval*, 2(3):137–215, 2008.

[146] C. Zhai and J. D. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of the 24th ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 334–342, 2001.

[147] C. Zhai and J. D. Lafferty. Two-stage language models for information retrieval. In *Proceedings of the 25th ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 49–56, 2002.

[148] C. Zhai and J. D. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22(2):179–214, 2004.

[149] B. Zhang, H. Li, Y. Liu, L. Ji, W. Xi, W. Fan, Z. Chen, and W.-Y. Ma. Improving web search results using affinity graph. In *Proceedings of the 28th ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 504–511, 2005.

[150] J. Zhang, J. Tang, and J.-Z. Li. Expert finding in a social network. In *DASFAA*, pages 1066–1069, 2007.

[151] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *Advances in Neural Information Processing Systems*, 2003.

[152] D. Zhou, E. Manavoglu, J. Li, C. L. Giles, and H. Zha. Probabilistic models for discovering e-communities. In *WWW*, pages 173–182, 2006.

[153] D. Zhou, B. Schölkopf, and T. Hofmann. Semi-supervised learning on directed graphs. In *NIPS*, 2004.

[154] X. Zhu. Semi-supervised learning literature survey. *Technical report, University of Wisconsin-Madison*, 2006.

[155] X. Zhu and Z. Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical report, 2002.

[156] X. Zhu, Z. Ghahramani, and J. D. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International Conference on Machine Learning*, pages 912–919, 2003.

[157] X. Zhu and J. D. Lafferty. Harmonic mixtures: combining mixture models and graph-based methods for inductive and scalable semi-supervised learning. In *ICML*, pages 1052–1059, 2005.