The Pennsylvania State University
The Graduate School

MINING FEEDBACK IN RANKING AND RECOMMENDATION SYSTEMS

A Dissertation in
Information Sciences and Technology
by
Ziming Zhuang

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

May 2009

The dissertation of Ziming Zhuang was reviewed and approved* by the following:

C. Lee Giles
David Reese Professor of Information Sciences and Technology
Professor of Computer Science and Engineering
Co-Chair of Committee and Dissertation Co-Advisor

Prasenjit Mitra
Assistant Professor of Information Sciences and Technology
Co-Chair of Committee and Dissertation Co-Advisor

Dongwon Lee
Associate Professor of Information Sciences and Technology

Wang-Chien Lee
Associate Professor of Computer Science and Engineering

John Yen
University Professor of Information Sciences and Technology
Associate Dean for Research and Graduate Programs

*Signatures are on file in the Graduate School.

Abstract

The amount of online information has grown exponentially over the past few decades, and users become more and more dependent on ranking and recommendation systems to address their information seeking needs. The advance in information technologies has enabled users to provide feedback on the utilities of the underlying ranking and recommendation systems, and in return the systems to utilize such feedback for enhanced service. It is increasingly important to be able to tailor relevant information for different users and applications given various feedback. In this dissertation, we study how feedback can be utilized to improve the service quality of ranking and recommendation systems, in the application context of Web search engines and large scale digital libraries.

We first introduce a flow-based collaborative ranking model of users' collective feedback in an online information seeking process. The model constructs a flow-based network to describe the relationship among collaborating users, queries, and documents. This generic model allows us to quantitatively investigate the properties of a collaborative ranking process. We also present a collaborative ranking algorithm derived from this model.

We then study the implicit user feedback in query reformulations in the context of general purpose Web search ranking. Specifically, we apply the knowledge of user feedback to address the problems introduced by the *under-specified* queries. We propose an algorithm to leverage the *query context* to refine the relevance ranking of the search results. We describe empirical evaluations which demonstrate the benefits of our proposal.

We then study the utility of feedback in two vertical ranking and recommendation systems. The first is a geographic information retrieval system. We analyze users' historical clicks as their implicit feedback to the system, and study two click-based models to infer geographical preference based on mining the user click stream data. We are able to identify search queries and documents with spatial

specificity, and generate effective relevance features for search ranking.

The second vertical is a venue recommendation system for digital libraries. We study the feedback loop of publication quality and venue organization, and propose a set of heuristics to automatically discover prestigious (as well as low-quality) publication venues by exploring the characteristics of the venue organizers.

Table of Contents

List of Figures

List of Tables

Acknowledgments

First and foremost, I would like to thank everyone who has helped me with his or her suggestions and support during my time at the Pennsylvania State University. Especially, I would like to thank my dissertation committee members, Prof. C. Lee Giles, Prof. Prasenjit Mitra, Prof. Dongwon Lee, and Prof. Wang-Chien Lee for their moral and material support, valuable and helpful insights, and the countless time and efforts spent guiding my thesis research.

Second, I would like to extend my thanks to my internship mentors in the industry: Silviu Cucerzan, Marc Najork, and Eric Brill from Microsoft Research, and Cliff Brunk from Yahoo! Labs. It was with their support that I had the opportunity to work on large-scale search engine logs.

Third, I would like to express my sincere appreciations to my parents, Jiebing Li and Daorong Zhuang, whose love and confidence in me has accompanied me since birth.

Last but not the least, I would like to thank my wife, Ying Shi, for the warm encouragement and enduring love she has offered, which made it possible for me to reach this point in my life.

Chapter 1

Introduction

In the past few decades we have witnessed the exponential growth of information available online. The growth rate is getting faster on the Web [2, 52, 79], owing to enabling technologies for self-publishing and the recent flourish of user generated content such as blogs. Studies in 2007 estimated the number of Web pages to be close to 30 billion [2], and the number of URLs found by a major commercial major search engine is even higher at one trillion[1] [3]. To make the situation worse, although there are a number of standardization bodies such as the W3C[2] and IETF[3], there is no *"quality control" per se* for the online content. Thus, information-seeking users have to face a prohibitive situation similar to *"finding needles in a haystack"*.

There are a number of directions along which solutions to this problem have been proposed. For example, the Semantic Web is proposed as an extension of the *raw* Web: by adding an additional layer on which the semantics of information and services on the Web is annotated, it enables computers to accurately understand and utilize these information and services to serve the users. However, until the Semantic Web becomes ubiquitous, the most widely adopted solutions are ranking and recommendation systems operating directly on the *raw* Web.

What are ranking and recommendation systems? Before we give them formal definitions in Chapter 2, here we describe them intuitively as follows. In an

[1] According to Google, not all lead to unique Web pages.
[2] The W3C Consortium http://www.w3c.org/
[3] The Internet Engineering Task Force, http://www.ietf.org/

information-seeking scenario where the amount of available information is so huge that a user cannot practically evaluate, comprehend, and consume all of them, a ranking and recommendation system filters irrelevant information, ranks the different pieces of information based on a number of user-defined metrics, recommends and presents a much smaller subset of information to the user. Applications of these systems, such as a Web search engine, have become the indispensable information gateway for users nowadays [1].

In this dissertation, we study how feedback can be utilized to improve the service quality of ranking and recommendation systems. Feedback is a process in which some proportion of the output signal is passed back as input, in order to dynamically adjusts the system's behavior. In the context of ranking and recommendation systems for online information, we are interested in two types of feedback (formal definitions to follow in Chapter 2):

- **Internal feedback.** feedback that propagate among the distinct entities internal to the system, such as two different but *somewhat correlated* documents.
- **External feedback.** feedback that propagate between the system and the external entities, such as a search engine and its users.

As in other systems and services that involve user interactions, external feedback provide a valuable source of user preference for ranking and recommendation systems. A typical scenario where user feedback help improve system performance is collaborative search ranking, in which the collective usage patterns of collaborating users are analyzed and utilized as implicit feedback to enhance the relevance ranking quality. Thus, our first research question is to investigate and model such a collaborative ranking process, more specifically,

How do we model the collaborative search process, formulate a quantitative representation of the implicit feedback from the collaborating users, and derive a feedback-based collaborative ranking framework?

In the first part of this dissertation, we systematically investigate the properties of various feedback among users, documents, and queries in a collaborative ranking process. We propose a flow-based collaborative ranking model, which casts the collaborative ranking problem into a network flow problem. The relations among the heterogeneous entities - queries, documents, and collaborating users - are drawn into a concise and cohesive framework. Based on this model, we describe *FlowRank*, a collaborative ranking algorithm, and demonstrate its utility in improving relevance ranking.

The next research question is, from a practitioner's standpoint, to study how feedback can be used in real-world ranking and recommendation systems, more specifically,

How do we utilize feedback in the application context of general purpose Web search ranking systems, as well as vertical (i.e., special purpose) search ranking and recommendation systems?

In the second part of this dissertation, we first study an aspect in the flow-based collaborative ranking model: the utility of users' feedback on search queries in a general purpose Web search ranking system. We describe a method *Q-Rank* to mine the collective feedback embedded in query logs, and to use the distributional information of what we defined as *query contexts*, to effectively improve the relevance ranking for search queries that are not very well articulated.

We then present two case studies on the utility of feedback in special purpose ranking and recommendation systems. First, we investigate users' clicks as their implicit feedback on locality preference in geographic search ranking. By mining historical user click stream data, we present two models of users' geographical interests, and address three important issues in spatial Web search. First, search queries and documents can be classified by the model according to their spatial specificity. Second, the geographical centers of interests for queries and documents can be inferred. Finally, the model is adapted to generate meaningful relevance features for search ranking. Second, we study feedback in the context of a venue recommendation system in digital libraries. Specifically, we investigate the correlation between the publishing and collaboration patterns of the program committee

members, and the quality of the publication venues. Based on these feedback, we propose a number of heuristics to automatically discover prestigious as well as low-quality academic conferences. See Figure 1.1 for an overview of the structure of this dissertation.

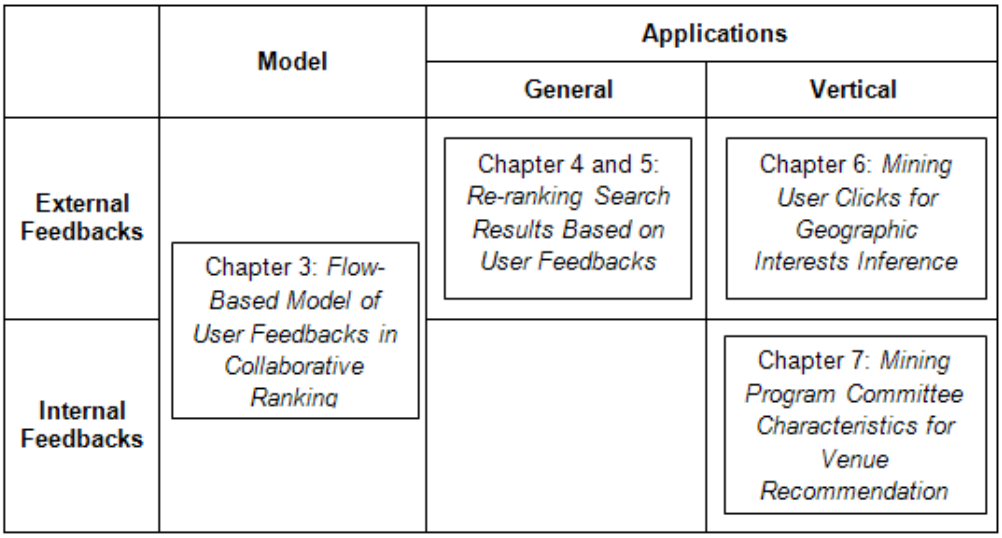

Figure 1.1. Overall structure of the dissertation and the relationship among different chapters.

Key Contributions

1. We introduce a flow-based model of collaborative ranking [140], which quantitatively describes the various feedback relationship among users, queries, and documents. This model translates the collaborative ranking problem into a network flow calculation problem. We also present a systematic case study on a number of ranking scenarios using the model.

2. We study the utility of feedback in general purpose Web search ranking systems. We propose a method *Q-Rank* to leverage the implicit feedback on search queries and to apply such feedback to effectively refine the ranking of Web search results, especially for naive or ambiguous queries [138, 139].

3. We demonstrate the utility of user clicks as implicit feedback in a geographic search ranking system. We describe a method to use the geographic distribution of user clicks to model the collective locality preference [137, 136].

4. We investigate the feedback relationship between the academic venues, the organizers, and their publishing and collaboration patterns. We propose a set of heuristics to rank and recommend academic conferences, and to discover emergent venues of good quality [141].

Organization

The rest of this dissertation is organized as follows. In Chapter 2, we present a detailed review of relevant literature and provide definitions for common terminologies and annotations used throughout the dissertation. In Chapter 3, we introduce the flow-based collaborative ranking model, *FlowRank*, systematically study the feedback relationship among collaborating users, search queries, and relevant documents. We also empirically demonstrate the effectiveness of our model. In Chapter 4, we continue to explore one specific type of feedback, study the utility of users' implicit feedback on search queries, and describe a method *Q-Rank* to improve relevance ranking based on such feedback. We further evaluate our *Q-Rank* proposal, comparing it against the query-log based authority analysis algorithm [84], and describe our evaluation in Chapter 5. While Chapters 3, 4, and 5 are for general purpose search ranking, Chapters 6 and 7 present two case studies on the utility of feedback in vertical ranking and recommendation systems. In Chapter 6, we investigate two models of users' geographic interests, based on historical click patterns as the implicit feedback. The models we proposed prove effective in identifying search queries with spatial specificity, and in improving relevance ranking for geographic search queries. In Chapter 7, we study the problem of venue ranking and recommendation, by investigating the feedback relationship of a venue and its organizers. We describe a method to measure the quality of academic conferences. In Chapter 8, we conclude the dissertation.

Chapter 2

Background and Related Work

2.1 Preliminaries

2.1.1 Common Terminologies

We define the common terminologies used throughout this dissertation as follows.

Web. The Web, or the World Wide Web, is a system of interlinked hypertext documents (Web pages) identified by Uniform Resource Identifiers (URI) and accessible via the Internet [9]. We denote the Web by $\mathbb{W}$, and a Web page $\mathbf{w} \in \mathbb{W}$.

Document. Without loss of generality, we define a document as the atomic unit for ranking and recommendation. Depends on the application context, a document can be a Web page, an academic paper, a publication venue, etc. We denote the set of documents to be ranked or recommended as $\mathbb{D}$, and a document $\mathbf{d} \in \mathbb{D}$.

Query. A query is a description of an information need. In existing literature, a query can be represented as a vector in a term/concept space [110], a state in a process [13], a set of topics [16], etc. In the context of this dissertation, we denote a query as $\mathbf{q}$ which is a length-K sequence of one or more query terms $t_1 t_2 \dots t_k$.

User. A user typically refers to a human being who has an information need and interacts with the ranking and recommendation system with the goal to satisfy the need. During the interaction process, the user could provide feedback to the system, and the system can adjust its behavior in order to better serve the user. Note that in practice, not all users provide truthful and beneficial feedback, and some are even malicious [33]. We denote the set of users by $\mathbb{U}$, and a user $\mathbf{u} \in \mathbb{U}$. In the context of collaborative ranking and recommendation, we use the terms *user* and *collaborator* interchangeably.

Feedback. Feedback usually refers to a process in which some proportion of the output signal is passed back as input in order to dynamically adjusts behavior. In this dissertation, we are interested in two types of feedback within the context of ranking and recommendation systems. *Internal feedback* denotes the feedback that propagate *inside* the system, among queries and documents. An example of such feedback is the correlation between two *somewhat similar* documents. *External feedback* refers to the feedback that propagate between the system and its users, and is sometimes referred to as the *relevance feedback* [109]. We denote the *internal feedback* by $\mathbb{F}_i$, and the *external feedback* by $\mathbb{F}_e$.

2.1.2 Ranking and Recommendation

We use the terms *ranking* and *relevance ranking* interchangeably. Without loss of generality, the problem of *relevance ranking* can be defined as follows.
Relevance Ranking:

- Given a query $\mathbf{q}$ and a document $\mathbf{d}$, we assign a relevance score function $rel(\mathbf{q}, \mathbf{d}) \in [0, 1]$ according to criteria $\mathcal{C}$.
- Given a query $\mathbf{q}$ and two documents $\mathbf{d_i}$ and $\mathbf{d_j}$, $rel(\mathbf{q}, \mathbf{d_i}) > rel(\mathbf{q}, \mathbf{d_j})$ if and only if document $\mathbf{d_i}$ is more relevant w.r.t. query q than document $\mathbf{d_j}$.

Thus, the problem of *recommendation* can be defined as follows.
Recommendation: Given a query $\mathbf{q}$, a set of documents $\mathbb{D} = \{\mathbf{d}\}$, and a scoring function $rel(\mathbf{q}, \mathbf{d}) \in [0, 1]$, return a subset of documents $\mathbb{D}' \subseteq \mathbb{D}$, such that $\forall \mathbf{d} \in \mathbb{D}', rel(\mathbf{q}, \mathbf{d}) \geq \zeta$.

Trivially, when $\zeta = \max rel(\mathbf{q}, \mathbf{d})$, the recommendation system returns the most relevant document(s).

In the application context of Web search engines, users usually expect to find relevant information in the top-ranked search results, and more often than not they only look at the document snippets in the first one or two result pages [57]. On the other hand, highly-ranked documents have greater visibility, which usually translates into getting more attention and eventually leads to popularity [92]. Hence, the quality of ranking and recommendation systems potentially introduces a huge impact on the users' perception of information on the Web, and is the critical, closely-guarded core component of search engines.

Given its importance, there has been a surge of research interests in ranking and recommendation algorithms in the past few decades. We'll survey the literature in the second half of this chapter.

2.1.3 Applications of ranking and recommendation systems

Web search engines and digital libraries probably have the most prevalent applications of ranking and recommendation systems. There are different types of search engines. For example, a special purpose (a.k.a. vertical or topical) search engine retrieves information in a specific domain (e.g., finding products or people), in contrast to general purpose search engines such as Google[1] or Yahoo[2]. An enterprize search engine indexes and searches the documents typically within the intra-network of an organization, in contrast to a Web search engine which freely traverses the World Wide Web. There are also multimedia search engines that are specifically designed to search images, audio and video files. In this thesis we limit our scope of discussion to Web search engines which retrieves online documents (i.e., webpages), and from now on we will use the terms *documents* and *webpages* interchangeably.

Web search engines harvest, index, and archive online contents (e.g., documents, images, and videos), and provide a retrieval service to the users. The typical architecture of a query-based Web search engine consists of an injection pipeline (a crawler module and an indexer module) and a retrieval pipeline (a

[1] http://www.google.com
[2] http://search.yahoo.com

query parser module, a searcher component, and a ranking module), as shown in Figure 2.1. In the injection pipeline, the crawler module harvests online documents from the Web, and the documents are then indexed by the indexer module. The indexes are stored and maintained by the search engine, usually in a distributed and layered fashion. In the retrieval pipeline, when a user issues a search query, the query parser module parses the user query, converts it to an internal representation of the query, and pass this *internal query* to the searcher module. The searcher module selects from the indexes the contents based on the *selection criteria*, and then the ranking module ranks these contents according to the *ranking criteria*. Eventually, the search engine returns a ordered list of documents to the user. Note that the criteria used in the *selection phase* and the *ranking phase* can be different, depending on a number of metrics (e.g., relevance to the user query, freshness, and popularity). These modules are the essential building blocks of contemporary Web search engines.

In practise, search engines are usually implemented with many more auxiliary modules. For example, a *deduplication module* is installed to filter duplicate contents, and a *caching module* is typically built into the retrieval pipeline to speed up the process for repeated popular queries. The indexes are usually *layered*: the indexes of popular or high-quality contents are often separated from the indexes of the remainder, optimizing the retrieval performance of more relevant contents.

2.2 Relevance Ranking

2.2.1 Basic Relevance Models

We first briefly introduce two basic relevance models: the Boolean model and the vector model.

Based on Boolean algebra and set theory, the Boolean model is the simpler model of the two. The model considers each term in a document as either present or absent, thus let $d = (t_1, t_2, ..., t_k)$ denote a document $\mathbf{d}$ with k index terms, and let $weight_i(d)$ denote the weight of a term t_i in $\mathbf{d}$: either 1 (present) or 0 (absent). On the other hand, the model represents a search query as a disjunction of conjunctive vectors. Let q_{DNF} denote the disjunctive normal form of a query $\mathbf{q}$,

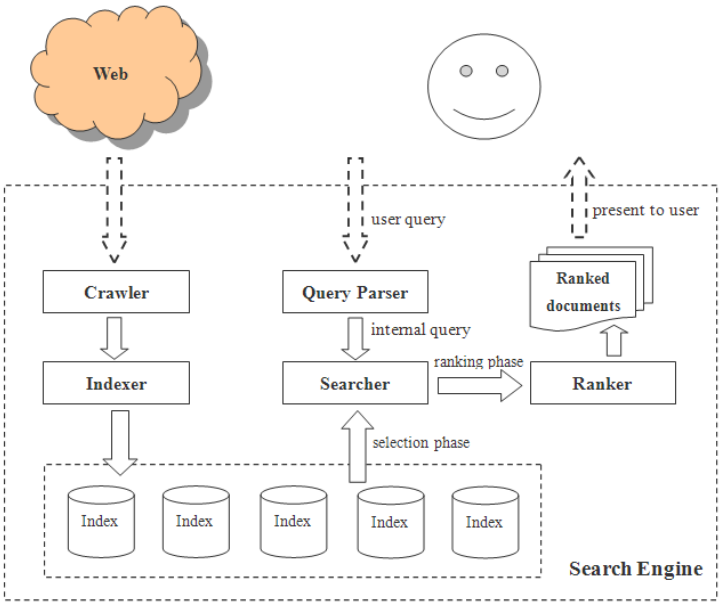

Figure 2.1. The typical architecture of a Web search engine consists of an injection pipeline (a crawler module and an indexer module) and a retrieval pipeline (a query parser module, a searcher component, and a ranking module).

and let $q_{CC} \in q_{DNF}$ denote a conjunctive component, the relevance of a document $\mathbf{d}$ to a query $\mathbf{q}$ is defined as

$$rel(\mathbf{q}, \mathbf{d}) = \begin{cases} 1, \ \exists q_{CC} | (q_{CC} \in q_{DNF}) \wedge (\forall i, weight_i(d) = weight_i(q_{CC})) \\ 0, \text{otherwise.} \end{cases}$$

Intuitively, the Boolean model considers a document as *relevant* if and only if there is an exact match between the document terms and the query terms. There is no *partial matching* per se, and each term t_i is treated equally.

In contrast, the vector model assigns non-binary weights to the terms. Let $t_1, t_2, ..., t_k$ denote the k terms in the relevance model, a k-dimension vector $d = (w_1, w_2, ..., w_k)$ denote a document $\mathbf{d}$, with each dimension w_i corresponding to the weight of a term t_i in $\mathbf{d}$. Further, let another k-dimension vector $q = (w_1^q, w_2^q, ..., w_k^q)$ denote a query $\mathbf{q}$, and each dimension w_i^q denote the weight of a term t_i in $\mathbf{q}$. The relevance of a document $\mathbf{d}$ to a query $\mathbf{q}$ is defined as

$$rel(\mathbf{q}, \mathbf{d}) = \frac{q \cdot d}{|q| \times |d|} = \frac{\sum_{i:1 \to k} w_i \times w_i^q}{\sqrt{\sum_{i:1 \to k} (w_i)^2} \times \sqrt{\sum_{i:1 \to k} (w_i^q)^2}} \tag{2.1}$$

One of the many ways to calculate the weight w_i of a term t_i is the Term-Frequency Inverse-Document-Frequency (tf-idf) scheme [110], which computes w_i in $\mathbf{d}$ as

$$w_i = freq_d(t_i) \times \log \frac{N}{n_{t_i}} \tag{2.2}$$

where $freq_d(t_i)$ denotes the frequency of term t_i in document $\mathbf{d}$, N denotes the total number of documents in the indexes, and n_{t_i} denotes the number of documents in which t_i appears at least once.

2.2.2 Ranking Based on Language Models

The basic assumption behind relevance ranking algorithms based on language models [54, 103] is that when users search for webpages relevant to a topic, they may have hypothesized *what a relevant webpage may look like*, and generated the search queries from these hypothetical documents. For example, looking for information about New York City, a user could assume that any relevant webpage may contain the phrases *new york* and *manhattan*, and then uses these two phrases to compose a search query *"new york" manhattan* hoping to retrieve the relevant documents she/he has in mind. Based on this assumption, relevance ranking algorithms derived from language models rank documents by the probability that the language model behind each document can generate the query terms.

Formally, given a document $\mathbf{d}$ and a query $\mathbf{q} =< T_1 T_2 ... T_n >$, T_i denotes a query term, the probability of $\mathbf{q}$ generated from $\mathbf{d}$'s language model is

$$\begin{aligned} rel(\mathbf{q}, \mathbf{d}) &= P(D) \cdot P(Q|D) \\ &= P(D) \cdot P(T_1, T_2, ..., T_n|D) \\ &= P(D) \cdot \prod_{i=1}^{n} [(1 - \lambda_i) P(T_i) + \lambda_i P(T_i|D)], \end{aligned} \tag{2.3}$$

in which

$$P(D = \mathbf{d}) = \frac{\sum_t tf(t,d)}{\sum_t \sum_d tf(t,d)}, \quad (2.4)$$

$$P(T_i = t_i) = \frac{df(t_i)}{\sum_t df(t)}, \quad (2.5)$$

$$P(T_i = t_i | D = \mathbf{d}) = \frac{tf(t_i,d)}{\sum_t tf(t,d)} \quad (2.6)$$

where $df(t)$ denotes the number of documents which contain term t, and $tf(t,d)$ denotes the term frequency of t in d.

Intuitively, the first part in Equation 2.3 represents the probability of a term in the corpus, and the second part represents the probability of the term appearing in the document. The unknown parameter λ_i determines the importance of a term t_i w.r.t. the relevance ranking model. For a given term, if $\lambda_i = 0$, the term has no impact on the ranking of the document, otherwise if $\lambda_i = 1$, the term is mandatory for the document to be considered relevant (i.e., documents which do not contain t_i are considered irrelevant). In practise, λ_i can be estimated as the percentage of relevant documents which also contain the term t_i, or it can be optimized iteratively using the Expectation-Maximization (EM) algorithm [36].

Comparing with the vector relevance model, ranking algorithms based on a language model have a desirable smoothing effect: even if a document does not contain some (or even all) of the query terms, it could potentially still be considered relevant for a number of reasons, e.g., the document contains the synonyms of the query terms. While the basic vector space model doesn't take into account synonyms, ranking algorithms based on language modeling have an advantage in being able to match the query terms with their synonyms. In essence, language models estimate $P(T_i = t_i | D = \mathbf{d})$, i.e., the likelihood of a document $\mathbf{d}$ generates an observed query term t_i. Thus, for a synonym query term t_i', $P(T_i = t_i' | D = \mathbf{d}) \simeq P(T_i = t_i | D = \mathbf{d})$ as the common prior $P(D = \mathbf{d})$ is drawn from the same distribution. Additionally, in Equation 2.3, the first part of the equation does not depend on the document prior but the probability of the query term occurring in the corpus. So a document might be missing some of the query terms, but still considered relevant if synonyms exist. Zhai and Lafferty compared several smoothing methods for language models in terms of information retrieval

performance [131].

A webpage typically contains various *fields*: URL, title, abstract, and a number of paragraphs in the document body. The match between the query terms and the document terms in different fields could have different contributions to the final relevance ranking of the document as a whole. The family of BM25 relevance models [102, 104], which became well-known due to the Okapi system [105] at TREC-3, take into account such differences, and assign different weights to the fielded matching scores. The matching scores are then combined in either a linear or a non-linear fashion to compute the final relevance ranking score of a document. One form of the BM25 scoring function [102] is

$$rel(\mathbf{q}, \mathbf{d}) = \sum_{t_i \in \mathbf{q}} qtf_{t_i} \cdot \frac{(k+1) \cdot tf_{t_i}}{k[(1-b) + b \cdot \frac{L_\mathbf{d}}{\overline{L_\mathbf{d}}}] + tf_{t_i}} \cdot w_i,$$

where qtf_{t_i} denotes the term frequency of a query term t_i in the query $\mathbf{q}$, tf_{t_i} denotes the term frequency of t_i in the document $\mathbf{d}$, $L_\mathbf{d}$ denotes the length of the document $\mathbf{d}$, $\overline{L_\mathbf{d}}$ denotes the average document length in the corpus, and w_i is the Robertson/Sparck-Jones weight [101]. k and b are two tunable parameters, for example, k was usually set to be between 1.2 and 2.0 in TREC, and b was often set at 0.75.

2.2.3 Ranking Based on Link Structure

Compared with the aforementioned relevance models which rely on the syntactic features of a document, link-based ranking models use the hyperlink structure among documents to derive the document ranking either globally or dependent on a given query.

The Web graph [11, 19] has been explored in a number of studies on the hyperlinked structure of the Web in the topic of web search [18, 22, 23]. PageRank [93] and HITS (*Hypertext Induced Topic Search*) [71] are two well-cited ranking models, which utilize information about the link structure of the Web. Both of them build upon the assumption that the quality of a webpage can be inferred by the number and the quality of pages linking to it, and compute the ranking scores iteratively. PageRank computes a global ranking score for all the documents on the Web independent of user queries, and does not take into account users'

topical interests. On the other hand, HITS works on a query-specific subgraph of the Web, so that the ranking scores are dependent on the user query, thus are more tailored to the users' interests. However, HITS normally requires more query-time processing and is also more susceptible to localized link spam. HITS computes ranking scores based on mutual reinforcement for two different types of webpages, *hubs* and *authorities*, and the *authority* webpages with higher ranking are considered more relevant.

A number of enhancements to these two algorithms have been proposed. In PageRank and HITS, all webpages are considered equally relevant at the beginning of the iteration. In contrast, the Hilltop algorithm [12] uses a set of *expert pages* that are already identified as authoritative in the topic domain of the query as the initial *relevant* webpages. Similarly, the topic-sensitive PageRank algorithm [53] pre-computes a set of topic-sensitive vectors according to the search query, and then uses these vectors to bias the PageRank computation towards the particular topic(s) relevant to the query.

SALSA [83] is an efficiency improvement to the original HITS algorithm. The main difference is that in SALSA the Web is represented as a bipartite graph: one side of the graph contains all the *hub nodes* (i.e., nodes with outlinks), and the other side contains all the *authority nodes* (i.e., nodes with incoming links). SALSA then performs the following two types of random walk on the bipartite graph. The *authority walk* is a walk from the authority side to the hub side and then back to the authority side of the graph. And the *hub walk* is exactly the opposite: a walk from the hub side to the authority side and then back to the hub side of the graph. Eventually, we can infer that the nodes on the authority side with the most visits are the top authorities, and those on the hub side with the most visits are the top hubs. Because the authority walk and the hub walk can be performed independently in parallel, SALSA is more efficient in terms of computation than the original HITS algorithm. Moreover, HITS was found to be more susceptible to the so-called emph"Tightly-Knit Community" effect, i.e., biased towards tightly-knit communities so that the pages within these communities were ranked higher than should be [83].

2.2.4 Collaborative Ranking

Collaborative ranking is in a way similar to meta-search: it leverages naive and advanced users, and re-ranks the initial search results based on the collective knowledge. Typical collaborative ranking algorithms can take into account the expertise level of the users through user profiling [26]. A study of the collaborative search process involves observation on the search behaviors of actual users collaborating as a team [120]. When a user is not satisfied with the search results, similar queries submitted by other users can be used to expand or improve the original query [130]. User clicks have been shown as an accurate reflection of their preferences on the retrieved pages [63, 64], to suggest search intentions [32] and similar queries and documents [128]. A hit matrix, which records users' clicks to pages, is implemented in the I-SPY system [47]. By using the collective click information as an indicator of the likelihood of a webpage being visited by the users, the system estimates the pages' relevance to a given query, and re-ranks the pages based on the learned preferences.

2.2.5 Ranking and Bibliometrics in Digital Libraries

Measuring the quality of publication venues is an important task in bibliometrics. The most widely adopted method to this task is to use Garfield's Impact Factor (IF) [49]: the average number of times the published papers are cited up to two years after publication. Since the introduction of the IF, it has been heavily criticized primarily for its sole dependence on citation counts [107], and therefore many alternatives, e.g., H-index [55], PageRank-like measure [15], and download-based measures [14], have been proposed to rank computer and information science journals [68, 89]. Several citation-based metrics have been proposed for ranking documents retrieved from a digital library [76], and to measure the quality of a small set of conferences and journals in the database field [10, 99]. A recent study [85] introduces topic modeling to further complement the citation-based bibliometric indicators, producing more fine-grained impact measures. Recently, several studies [90, 7, 40] have analyzed the scientific collaboration networks in different disciplines.

2.3 Geographic Information Retrieval

We review two major groups of studies that are related to our work. The first group focuses on geo-mapping search queries and Web resources, and detecting and understanding the "locations of interests".

A study of the 2001 Excite search engine query logs revealed the linguistical features of the search queries that contain geographic terms [112]. The results indicated that a significant portion of the search queries were geographically related, that geographic queries were typically longer than queries of other types, and that some queries could be ambiguous (e.g., more than one place with the same name). A more recent follow-up study by the authors who examined the query log collected over four weeks from a different search engine confirmed that geographic terms were not only prevalent in search queries, but also frequent in repeated queries terms [111].

Gravano et al. evaluated a number of classification models based on features extracted from the search results (e.g., location names found in the Web pages) to classify a search query as of global or local interests [51].

Online resources can be associated with three types of locations [125]: the *provider location*, which is the physical location of the service/content provider, the *content location*, which indicates the geographic location that the content is about, and the *serving location*, which is the geographical scope of the target audience. Of particular interests to this study are the last two types of locations. Wang et al. described in detailed methods to identify these three types of locations for Web resources [124]. Zhang et al. described two methods to detect the *geographical serving area* of websites [132]. The *serving area* of a website is defined by the geographical scope of the website's target audience, e.g., a website about Disneyland might be of interests to users all over the globe. One method was to infer the locations based on the IP addresses from which the user clicks originated: the more user clicks on a website come from one location, the more likely it is the location that is the main serving area of the website. The second method was to collect all the historical query terms for a website, and to derive the main serving location of the website from these terms using algorithms similar to those proposed by Wang et al. [126]. Amitay et al. described a system for tagging Web pages

with geographic locations based on identifying known placenames in the pages [5]. Their study also presented heuristics for disambiguating placenames, and a *focus-finding* algorithm intended to identify the primary region out of all the locations mentioned.

Different websites target the audience of various geographical scopes, some broader or smaller than the others. An earlier study proposed a quantitative definition of the center and the spread of the scope of a website, based on the geographical distribution of the textual references and the hyperlinks pointing to it [37].

Similarly, for geographic queries users typically expect the search results to be both *relevant* and *local*, i.e., within a certain radius (i.e., *neighborhood*) of a location. In a recent study, the authors described the mathematical representation of the notions of the *center* and the *extent* of neighborhoods [115].

Wang et al. described a method to detect the *dominant location* of a query [126]. The *dominant location* of a query is the query's (implicit or explicit) location agreed by the majority of users who know the answer to this query. The authors considered three information sources in descending order of importance. The first source is the location inferred from the query terms. The second source is the location derived from user's IP address as well as the webpages that were previously clicked on for the same query. And the third and final source is the location names extracted from the search results for the query. The authors evaluated the three sources separately and in various combination, and demonstrated the advantage of their proposal in identifying the query locations over a dictionary lookup method and searching on Google.

There is also a significant line of work on visualizing geo-referenced data on the Web [4, 69, 70]. Ahern et al. described an analysis of geo-tagged images on Flickr[3], a vastly popular online photo hosting service, to correlate geolocations supplied by the photographers with the textual tags of the corresponding photos [4]. While this study was not about improving Web search, the proposed technique could be applicable to training a location identifier for search queries, based on the correlation between certain search terms and the geo-location coordinates.

The second group of studies focus on applying this knowledge to model the

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

spatial relationships (e.g., center and spread, closeness, overlap) and to improve Web information retrieval performance in general.

Delboni et al. proposed to directly exploit the positioning expressions in the natural language for geographical document retrieval [35]. Specifically, the authors extracted the spatial relationships among different geographic entities from the documents, and index the documents together with the corresponding entity and the *distance* to a landmark expressed by the spatial relationship. Thus, given a query which contains an entity and a landmark, the algorithm can retrieve documents about the relevant entity within a certain radius of the landmark.

Sheng et al. proposed a click-based model to discover the geographical patterns of the visitors' interests for websites, and studied the temporal change of these patterns [119]. Their work showed that the click data is a good indicator of users' various interests w.r.t. Web documents in specific geographical locations.

Users commonly include location names in their search queries to signal their locality preference. Zhang et al. studied the characteristics of such query rewriting with regard to geographical specificity [133]. In their follow up study, Jones et al. further investigated the distance between the user's physical location and the location specified in the search queries, and the variation in such distance for different query topics [66]. These two studies shed light on the user behavior of query reformulations containing location names.

Spatial ranking leverages the knowledge about the spatial relationship between the geographic region of the query and the candidate results. Larson and Frontiera evaluated several probabilistic spatial ranking algorithms with different weighting schemes for the overlapping region of the bounding boxes [77]. Lanfear described a ranking algorithm based on spatial overlay of the bounding boxes, and suggested that using the bounding box representation could lead to a lower precision than the bounding polygon representation [75]. Lee et al. exploited several heuristics-based improvements to the retrieval performance of handling range queries using the bounding box representation [80, 81].

Cai described at high level a spatial information retrieval model [21], integrating the notion of *distance* represented in the coordinate-based geographical space and the notion of *relevance* represented in the keyword-based vector space.

2.4 Use of Feedback in Relevance Ranking

Query expansion is an effective way to revise the original search query and improve the relevance ranking. Feedback-based query expansion [87] uses the documents retrieved initially as a source for relevance feedback. For example, Cui et al. mined click-through records of search results to establish mappings from query terms to strongly correlate document terms, and then used these document terms for query expansion [31]. Kraft and Zien proposed to generate a list of weighted query expansions for queries from the anchor texts of the retrieved documents [74]. These feedback-based query expansion algorithms are in contrast to thesauri-based query expansion algorithms [34, 94], which usually rely on corpus statistics to learn correlations among query terms.

Relevance ranking can also be refined during an interactive process involving the search engine user, in an iteration of search-and-feedback cycles [109]. Improving relevance ranking based on explicit user feedback require users to make explicit endorsements of the documents' relevance. Thus, one major disadvantage of such methods is that users could be quite reluctant to offer explicit feedback due to the extra efforts required. Furthermore, a system based on explicit feedback is generally easier to spam. On the other hand, implicit user feedback can be collected unobtrusively by analyzing users' search history [48], query reformulations [97], and exploiting click-through patterns [62, 63, 65, 98, 129, 135]. The quality of relevance ranking can also be improved by collaborative filtering algorithms to take into account the preferences of similar users [121]. Recent studies exploit query logs to refine search results. Past sequence of queries are used to complement the current query in estimating document relevance [118]. Given a user query, past queries are selected based on the similarity between their search results, and then are used to suggest an extended document list [88]. A formal user model is proposed based on the immediate search context for personalized ranking [117]. Preceding queries in the same search session are used to expand the current query. Recent developments try to learn the optimal relevance ranking based on the user's preferences [20, 45, 116].

Chapter 3

Flow-Based Model of User feedback in Collaborative Ranking

3.1 Introduction

In this chapter, we introduce a flow-based collaborative ranking model of users' collective feedback in online information seeking process. The model constructs a flow network to describe the relationship among collaborating users, queries, and documents. This general model allows us to quantitatively investigate the properties of collaborative ranking in more concrete terms.

Why does collaborative ranking work? In the context of Web search, the intuition behind collaborative ranking is that collective knowledge enhances the ranking of search results, and the expertise of advanced users help improve the search experience of naive users. The relevance ranking of search results for underspecified queries can be improved by exploiting similar and more articulated queries (and their corresponding search results) from other users. Learning from the collective access patterns on the search results, a collaborative ranking algorithm can favor certain documents. A number of methods to facilitate collaborative ranking have been proposed and some effectively implemented, particularly in online recommendation systems.

Previously, flow-based algorithms have been proposed to identify and mine online communities [41, 42], to perform clustering [29], and to identify the bottlenecks

in a Markov decision process [86]. In particular, Chitrapura and Kashyap proposed a flow-based model for document ranking, which uses the network flows in a search graph as a measurement of relevance [28]. In their model, the volume of the flows indicate the degree of relevance of the vertices (documents) to the associated labels (queries), and is used to compute a relevance ranking of the documents. While our model has a similar underlying idea, the two models are fundamentally different. Most notably, what we propose is a multi-user collaborative ranking model, while the Chitrapura and Kashyap model is a single-user model.

The rest of this chapter is organized as follows. We first introduce the flow-based model and discuss in detail our motivation, the graph transformation algorithm, and the *FlowRank* collaborative ranking algorithm. We then present and discuss the empirical evaluation of the model. We further exploit the model by considering ranking as a graph partitioning problem. Finally, we summarize this study, and also provide additional discussions on a number of ranking scenarios in the Appendix.

3.2 Motivation

Consider the following scenario of a typical collaborative search process: A user $\mathbf{u_a}$ searches the Web with query $\mathbf{q_o}$ *"Olympic national park"*, and retrieves a set of relevant documents $\mathbb{D}_o$. Then, suppose there is a collaborator $\mathbf{u_b}$, who searches with a similar query $\mathbf{q'}$ *"camping hiking Olympic"*, and retrieves another set of relevant documents $\mathbb{D}'$. We define *collaborators* to be those users who issue queries similar to $\mathbf{q_o}$ and retrieves relevant search results that are potentially interesting to $\mathbf{u_a}$ w.r.t. $\mathbf{q_o}$, regardless of whether they perform the search in a synchronous or asynchronous manner, or whether they are aware of each other. A collaborative ranking algorithm considers both how similar $\mathbf{q'}$ is to $\mathbf{q_o}$, and how similar $\mathbf{u_b}$ is to $\mathbf{u_a}$ in terms of interests and preferences. If $\mathbf{u_b}$ frequently accesses (e.g., clicks on) some of the retrieved documents, it could indicate that these documents are more relevant and preferable to other documents. Thus, $\mathbf{u_a}$ might also prefer these documents, on the condition that $\mathbf{q'}$ is somewhat similar to $\mathbf{q_o}$, and that $\mathbf{u_b}$ and $\mathbf{u_a}$ have similar search interests or preferences.

In the above scenario, we see that a model for collaborative ranking consid-

ers all of the following factors: the relevance between the original query $\mathbf{q_o}$ and the retrieved documents $\mathbb{D}_o$, the similarity between the user $\mathbf{u_a}$ and the collaborators $\mathbf{u_b}$, the similarity between the original query $\mathbf{q_o}$ and the similar queries $\mathbf{q}'$ issued by the collaborators, the relevance between these similar queries $\mathbf{q}'$ and their corresponding search results $\mathbb{D}'$, and the access pattern of all the users on the documents. Thus, given a query, each of these factors could contribute in a collaborative ranking algorithm to the relevance ranking of the retrieved documents.

In this study, our motivation (and goal) is to propose a formal model of collaborative ranking, in which all these factors are accounted for in a single framework that permits the study of the correlated search events. We accomplish this objective by exploiting the network flows of a transformed search graph. In our model, each retrieved document is represented as a sink vertex in the graph and associated with a flow value. The flow values are bounded by the different capacities of the arcs, which correspond to the various aforementioned factors (details to follow in the next section). When the flow network becomes saturated, the values of the flows associated with the documents are used to compute the collaborative relevance ranking.

3.3 The *FlowRank* Model

3.3.1 Graph Definition and Transformation

A search graph represents the relationships among different entities (queries, documents, and users) in a collaborative search process. We transform the search graph into a flow network in which arc capacities encode the relationships among these entities. In this section, we describe the algorithmic steps to derive the flow network model. An algorithm for collaborative ranking based on this model will be introduced in the next section.

First, we describe the structure of a search graph in which we represent the set of users, queries, and documents, denoted as $\mathbb{U}$, $\mathbb{Q}$, and $\mathbb{D}$ respectively. Furthermore, let $arc(m, n)$ denote a uni-directional arc from a vertex m to a vertex n. Such an arc $arc(m, n)$ exists between two vertices when the two vertices satisfy one of the following three conditions:

- $m \in \mathbb{U}, n \in \mathbb{Q}$, a user represented by m issues a search query represented by n.
- $m \in \mathbb{Q}, n \in \mathbb{D}$, a search query represented by m retrieves a document represented by n.
- $m \in \mathbb{U}, n \in \mathbb{D}$, a document represented by n receives a click from a user represented by m.

Assume that a target user $\mathbf{u^t}$ issues a target query $\mathbf{q^t}$, and retrieves a set of relevant documents $\mathbb{D}^t = \{\mathbf{d_i^t} | i = 1...m\}$ matching the target query. Let $\mathbb{Q}^t = \{\mathbf{q_i} | \exists j \in [1...n], arc(\mathbf{q_i}, \mathbf{d_j^t}) \neq null\}$ denote the set of all user queries which retrieve at least one of the documents in $\mathbb{D}^t$. Let $\mathbb{U}^t = \{\mathbf{u_i} | \exists \mathbf{q} \in \mathbb{Q}^t, arc(\mathbf{u_i}, \mathbf{q}) \neq null\}$ denote the set of users who submitted the query that retrieved at least one document in $\mathbb{D}^t$. Let G denote the whole search graph without constraints. We denote by G_t the subgraph of G with the vertices $\mathbb{D}^t \cup \mathbb{Q}^t \cup \mathbb{U}^t$ and all arcs incident in these vertices from G. Intuitively, the graph G_t represents those users who issue to the search engine the same query $\mathbf{q^t}$ or different (but similar) queries that retrieve at least one of the documents in $\mathbb{D}^t$. Neither the users who do not submit $\mathbf{q^t}$ nor the queries that do not retrieve any document in $\mathbb{D}^t$ are included in G^t, as the purpose is to rank collaboratively the documents in $\mathbb{D}^t$ by taking into account the relevant users, queries, and documents. Note that there are alternative definitions for $\mathbf{q_i} \in \mathbb{Q}^t$. For example, $\mathbf{q_i}$ can be defined as the expansion [123] of the target query $\mathbf{q^t}$, or a different query in the same search session (i.e., within a certain interval of time) as the query $\mathbf{q^t}$. Figure 3.1 (a) depicts a sample search graph G_t.

We then transform G_t to G'. Figure 3.1 (b) depicts G' constructed from G_t on the left. There are three types of capacities assigned to the arcs in G': type (a) represents the relevance between the target query and the documents; type (b) represents the degree of collective endorsement to the search results by the collaborators, either through the target query $\mathbf{q^t}$ or the alternative queries in $\mathbf{q^t}$; and type (c) represents the similarity between the targeted user $\mathbf{u^t}$ and his/her collaborators in $\mathbf{u^t}$.

We can now provide an intuitive justification for the transformation of G into G' (see Figure 3.2). A query represents the information needs of a user, and the motivation of search is to satisfy such needs. In G', flows of relevant information are

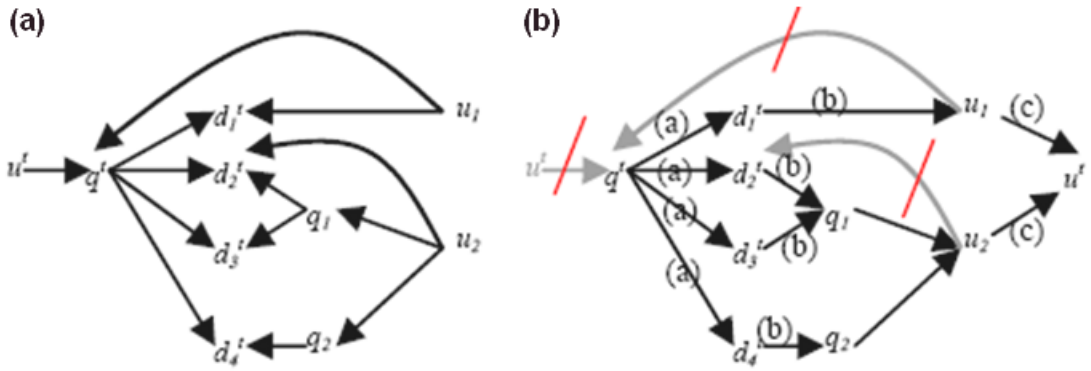

Figure 3.1. (a) A sample search graph G_t. (b) An example of G' transformed from G_t.

moving through a collaboration network towards the end user via different routes. On each of these routes, the flow contributes as a part to the overall relevant information gain received by the user. On each of the routes, there are an appropriately relevant document, an appropriately similar query, and an appropriately similar collaborator; all of them together determine the contribution (i.e., the value) of the corresponding network flow. As such, these flows, when saturated, represent the collaborative contribution to the overall satisfaction of the target user's information needs. By investigating G', we are able to study the relationship between the collaborative entities and sort the flows in the order of their contributions to the overall satisfaction of the target user's information needs. Accordingly the associated documents en route can also be ranked using the *FlowRank* algorithm described in the next section.

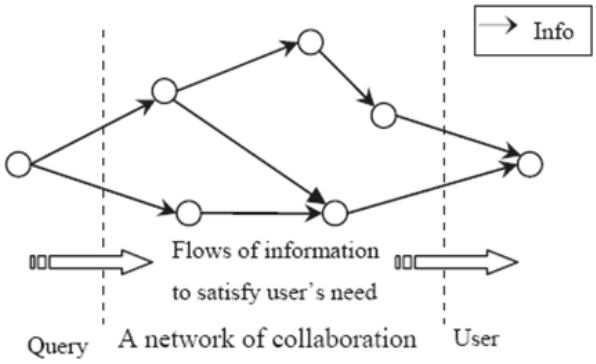

Figure 3.2. Intuitive justification of transforming G^t into G': Flows, when saturated, represent the collaborative contribution to the overall relevant information gain of the target user.

Graph Transformation Algorithm Let $\mathbb{U}' = \mathbb{U}^t - \{\mathbf{u^t}\}$ denote all users in G_t except for the target user, and $\mathbb{Q}' = \mathbb{Q}^t - \{\mathbf{q^t}\}$ the set of all user queries in G_t except for the target query $\mathbf{q^t}$. We transform G into G' as follows:

1. Remove $arc(\mathbf{u^t}, \mathbf{q^t})$. The relationship between $\mathbf{u^t}$ and $\mathbf{q^t}$ will be implicitly represented by the flow values in the transformed network. See Figure 3.3.

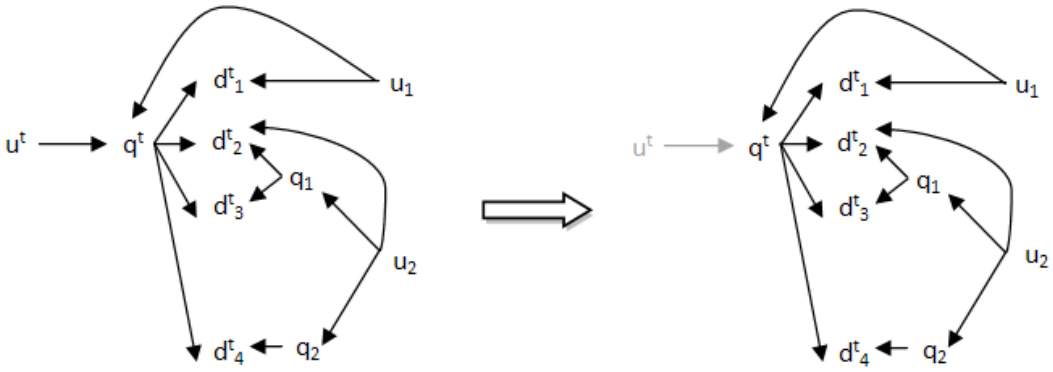

Figure 3.3. Graph transformation algorithm: Step 1

2. $\forall \mathbf{u_i} \in \mathbb{U}'$, add an arc $arc(\mathbf{u_i}, \mathbf{u^t})$, and define its capacity $C(\mathbf{u_i}, \mathbf{u^t})$ to be a similarity score between the two users, such that $0 < C(\mathbf{u_i}, \mathbf{u^t}) \leq 1$. This capacity places an upper-bound for the flow values proportional to the user-user similarity. See Figure 3.4.

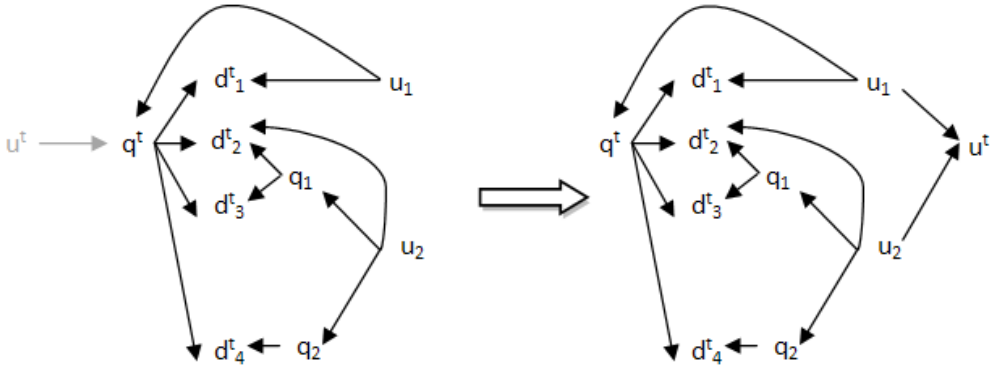

Figure 3.4. Graph transformation algorithm: Step 2

3. $\forall \mathbf{d_k} \in \mathbb{D}^t$, define the capacity $C(\mathbf{q^t}, \mathbf{d_k})$ of $arc(\mathbf{q^t}, \mathbf{d_k})$ to be a float value in $[0, 1]$. This capacity represents a relevance matching score between the

target query $\mathbf{q^t}$ and the document $\mathbf{d_k}$, and places an upper-bound for the flow values proportional to the query-document relevance.

4. $\forall \mathbf{u_i} \in \mathbb{U}'$ and $\forall \mathbf{q_j} \in \mathbb{Q}'$, if $arc(\mathbf{u_i}, \mathbf{q_j}) \neq null$ (i.e., the user $\mathbf{u_i}$ issued a query $\mathbf{q_j}$), reverse the direction of $arc(\mathbf{u_i}, \mathbf{q_j})$ to be $arc(\mathbf{q_j}, \mathbf{u_i})$, and assign its capacity $C(\mathbf{q_j}, \mathbf{u_i}) = 1$, so that the flow value on $arc(\mathbf{q_j}, \mathbf{u_i})$ is bounded only by the upstream capacities (i.e., the query-document relevance represented by the weights of $arc(\mathbf{q^t}, \mathbf{d_k})$), because of the reversed arc direction. Otherwise, set its capacity $C(\mathbf{q_j}, \mathbf{u_i}) = 0$. Note that the frequency of $\mathbf{u_i}$ issuing query $\mathbf{q_j}$ is not considered. See Figure 3.5.

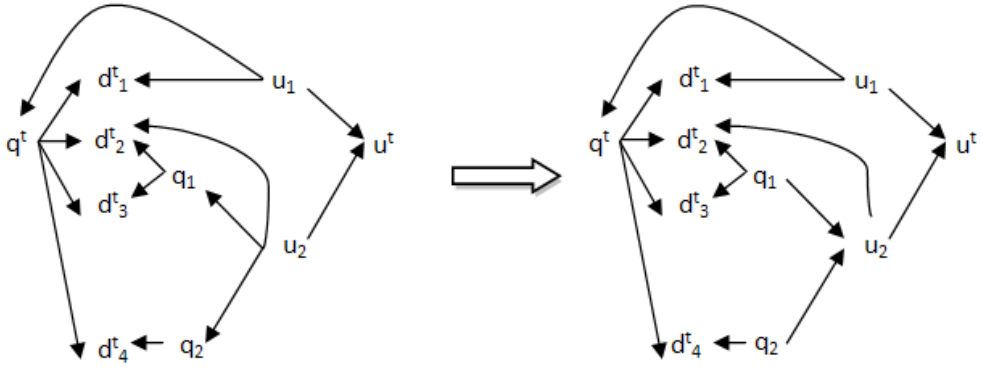

Figure 3.5. Graph transformation algorithm: Step 4

5. $\forall arc(\mathbf{q_j}, \mathbf{d_k})$ from a query $\mathbf{q_j} \in \mathbb{Q}'$ to a document $\mathbf{d_k} \in \mathbb{D}^t$, remove the arc $arc(\mathbf{u_i}, \mathbf{d_k})$, reverse the direction of the arc $arc(\mathbf{q_j}, \mathbf{d_k})$ and define its capacity $C(\mathbf{d_k}, \mathbf{q_j}) = sim(\mathbf{q_j}, \mathbf{d_k}) \cdot sim(\mathbf{q_j}, \mathbf{q^t}) \cdot P(\mathbf{d_k}|\mathbb{U}_{j,k})$, where $sim(\mathbf{q_j}, \mathbf{d_k}) \in [0, 1]$ represents a relevance matching score between the query $\mathbf{q_j}$ and the document $\mathbf{d_k}$; $sim(\mathbf{q_j}, \mathbf{q^t})$ represents how similar this alternative query $\mathbf{q_j}$ is to the target query $\mathbf{q^t}$; and $P(\mathbf{d_k}|\mathbb{U}_{j,k})$ indicates the conditional probability of users visiting webpage $\mathbf{d_k}$ given that they submit the query $\mathbf{q_j}$, which can also be rewritten as

$$C(\mathbf{d_k}, \mathbf{q_j}) = sim(\mathbf{q_j}, \mathbf{d_k}) \cdot sim(\mathbf{q_j}, \mathbf{q^t}) \cdot \frac{Click(\mathbb{U}_{j,k}, \mathbf{d_k})}{Click(\mathbb{U}_{j,k}, \mathbb{D}^t)} \tag{3.1}$$

where $\mathbb{U}_{j,k} = \{\mathbf{u_i} \in \mathbb{U}' | arc(\mathbf{u_i}, \mathbf{q_j}) \neq null \wedge arc(\mathbf{q_j}, \mathbf{d_k}) \neq null\}$ ($\mathbb{U}_{j,k}$ typically contains more than one user), $Click(\mathbb{U}_{j,k}, \mathbf{d_k})$ is the total number of clicks

made by the users who submit $\mathbf{q_j}$ on $\mathbf{d_k}$, $Click(\mathbb{U}_{j,k}, \mathbb{D}^t)$ is the total number of clicks of these users on the whole search result set.

$C(\mathbf{d_k}, \mathbf{q_j})$ places an upper-bound of the flow values jointly decided by the similarity between an alternative query $\mathbf{q_j}$ and the target query $\mathbf{q^t}$, the relevance matching score between $\mathbf{q_j}$ and the document $\mathbf{d_k}$, and the likelihood that a user visits $\mathbf{d_k}$ given that he/she submits $\mathbf{q_j}$. See Figure 3.6.

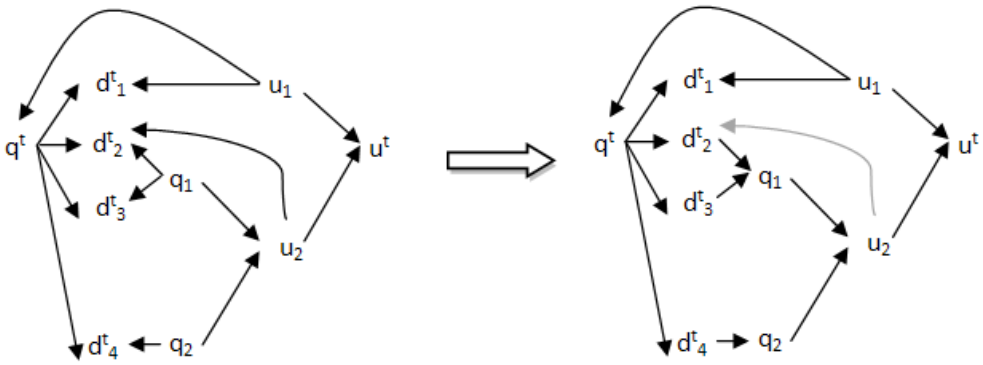

Figure 3.6. Graph transformation algorithm: Step 5

6. $\forall arc(\mathbf{u_i}, \mathbf{q^t})$ for which $\exists \mathbf{d_k} \in \mathbb{D}^t, arc(\mathbf{u_i}, \mathbf{d_k}) \neq null$, reverse the direction of $arc(\mathbf{u_i}, \mathbf{d_k})$ and assign $C(\mathbf{d_k}, \mathbf{u_i}) = sim(\mathbf{q^t}, \mathbf{d_k}) \cdot P(\mathbf{d_k}|\mathbf{u_i}) \cdot [1 - P(\mathbf{d_k}|\overline{\{\mathbf{u_i}\}})]$, where the last two factors represent the probability of a user $\mathbf{u_i}$ visiting $\mathbf{d_k}$, and the conditional probability of other users *not* visiting $\mathbf{d_k}$ given that they also submit the target query $\mathbf{q^t}$. We can also rewrite this as

$$C(\mathbf{d_k}, \mathbf{u_i}) = sim(\mathbf{q^t}, \mathbf{d_k}) \cdot \frac{Click(\mathbf{u_i}, \mathbf{d_k})}{Click(\mathbf{u_i}, \mathbb{D}^t)} \cdot (1 - \frac{Click(\overline{\{\mathbf{u_i}\}}, \mathbf{d_k})}{Click(\overline{\{\mathbf{u_i}\}}, \mathbb{D}^t)} \tag{3.2}$$

where $\overline{\{\mathbf{u_i}\}} = \mathbb{U}' - \mathbf{u_i}$, $Click(\mathbf{u_i}, \mathbf{d_k}) = 1$ if the user $\mathbf{u_i}$ clicks on document $\mathbf{d_k}$ and 0 otherwise, and $Click(\mathbf{u_i}, \mathbb{D}^t)$ denotes the number of documents clicked by $\mathbf{u_i}$.

$C(\mathbf{d_k}, \mathbf{u_i})$ places an upper-bound for the flow value jointly decided by the matching score between the target query qt and the document $\mathbf{d_k}$, and the likelihood that other users also visit $\mathbf{d_k}$ given that they also submit $\mathbf{q^t}$. See Figure 3.7.

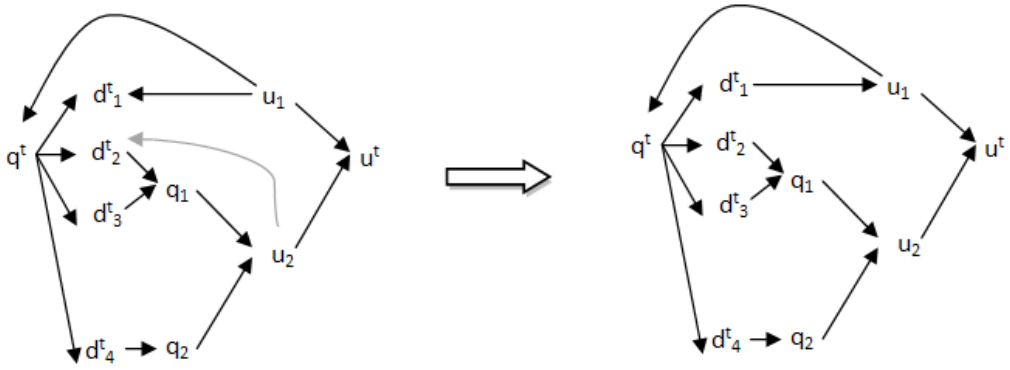

Figure 3.7. Graph transformation algorithm: Step 6

7. $\forall \mathbf{u_i} \in \mathbb{U}'$, Remove all the arcs $arc(\mathbf{u_i}, \mathbf{q^t})$. Because the relationship between these entities is captured in the definition of $C(\mathbf{d_k}, \mathbf{u_i})$, these arcs are redundant and do not contribute to the collaborative ranking process. See Figure 3.8.

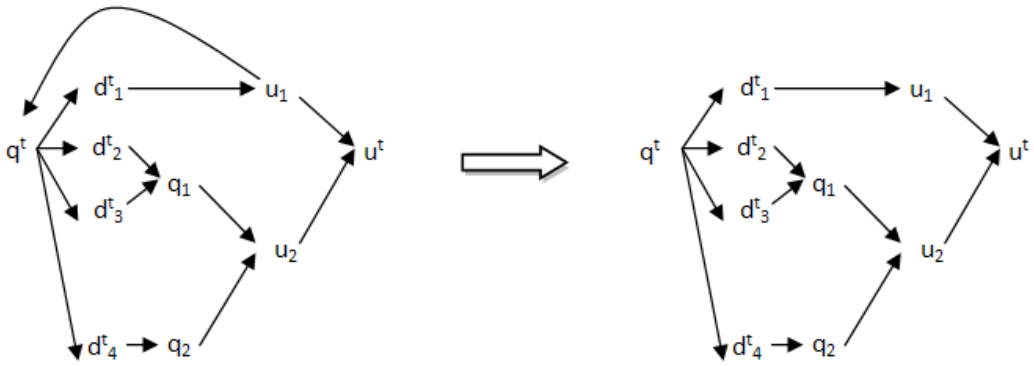

Figure 3.8. Graph transformation algorithm: Step 7

8. END

Remarks on the graph transformation algorithm:

- The algorithm reverses the directions of some arcs to build the cascade of upper-bounds for the network flows, i.e., type (a) capacities are more important than type (b), and type (b) are more important than type (c). This is because we should always prioritize the relevance between the target query and the search results, and consider the collective endorsement by other users

as an assistive measure. Please see the following algorithm for details on the reversion of the arcs' directions.

- Although the steps (2)-(6) of the following algorithm provide a specific computational recipe of the arc capacities, alternative definitions can also be used to weigh the arcs between the vertices.

3.3.2 The *FlowRank* Algorithm

Let s denote the source and t denote the sink, $F_{max}(s,t)$ is defined as the maximum flow that can be routed from s to t, obeying all the capacity constraints. Intuitively, if the arcs are water pipes, the vertices are where they join each other, and the capacities on the arcs represent the cross-sectional area of these pipes, to find the maximum flow is to find how much water can be moved from the source s to the sink t, given the constraints of the cross-sectional area of the pipes. Consider the transformed search graph in Figure 3.2, the maximum *information flow* represents how much relevant information gain one can obtain for the target user.

The *FlowRank* algorithm is described in Algorithm 3.3.2.

Algorithm 1 The *FlowRank* Algorithm

Require: Graph G'
Ensure: Permutation $\pi\{\pi(\mathbf{d_1^t}), ..., \pi(\mathbf{d_m^t})\}, \mathbf{d_1^t}, \ldots, \mathbf{d_m^t} \in \mathbb{D}^t$
  $F_{max}(\mathbf{q^t}, \mathbf{u^t})$ = the maximum flow from source $\mathbf{q^t}$ to sink $\mathbf{u^t}$
  for all document $\mathbf{d_i^t} \in \mathbb{D}^t$ **do**
    $F(\mathbf{q^t}, \mathbf{d_i^t})$ = the flow value from source $\mathbf{q^t}$ to sink $\mathbf{d_i^t}$
  end for
  Sort the documents $\mathbf{d_i^t} \in \mathbb{D}^t$ in descending order of $F(\mathbf{q^t}, \mathbf{d_i^t})$
  return $\{\pi(\mathbf{d_1^t}), ..., \pi(\mathbf{d_m^t})\}$

The *FlowRank* algorithm is based on the well-known *Maximum Flow - Minimal Cut Theorem* [43]. This theorem proves that the maximum flow of a given network is equal to the minimal cut that separates the source and the sink. In our model, the cut is $\mathbb{D}^t$. We proved it as follows:

Lemma 1. $\mathbb{D}^t = \mathbf{d_i^t}$ *is the set of cut-vertices of* G'.

Proof. A set of cut-vertices denotes the set of vertices whose removal will disconnect the network [24]. By assumption that $\mathbb{D}^t$ is *not* the set of cut-vertices of G',

we remove $\mathbb{D}^t$ and its related arcs, and still have $\mathbf{q^t}$ connected to G' via a set of vertices $\mathbb{V}_{qt}$. $\forall \mathbf{v_{qt}} \in \mathbb{V}_{qt} : \mathbf{v_{qt}} \notin \mathbb{Q}$, because, by definition, no arc exists between any two queries, i.e., $arc(\mathbf{q^t}, \mathbf{q_i}) = null$. On the other hand, $\mathbf{v_{qt}} \notin \mathbb{U}$ because of step (4) and (5) in the graph transformation algorithm in Section 4.3.1. Thus, $\mathbf{v_{qt}} \notin \mathbb{D}$. Because $\mathbb{D} = \mathbb{D}^t \bigcup \mathbb{D}'$ and $\mathbb{D}'$ is removed in step (1) in the graph transformation algorithm, $\mathbf{v_{qt}} \in \mathbb{D}^t$. But this contradicts the assumption that $\mathbb{D}^t$ has already been removed from G'. □

The proof of Lemma 1 is important for the validity of the algorithm, since $\forall F(\mathbf{q^t}, \mathbf{u^t}), \exists \mathbf{d_i^t} \in \mathbb{D}^t$ which is on the route of the flow $F(\mathbf{q^t}, \mathbf{u^t})$. Because the documents $\mathbf{d_i^t} \in \mathbb{D}^t$ are not inter-connected, we conclude the there is precisely one $\mathbf{d_i^t}$ on each of the routes of $F(\mathbf{q^t}, \mathbf{u^t})$, so that we can rank $\mathbf{d_i^t}$ by sorting the flow values $F(\mathbf{q^t}, \mathbf{d_i^t})$.

3.4 Evaluation

3.4.1 Experiment Setup

From the query logs of a popular scientific document search engine, the *CiteSeer Digital Library*[1], we randomly sampled 100 queries and the associated 1,334 unique documents retrieved by the search engine as relevant documents. Obtaining editorial ratings for a large number of documents is an extremely time-consuming and labor-intensive process, however the size of our dataset is comparable to those previously used [114]. The sampled queries were anonymized, and then verified by human annotators to be meaningful [100].

To evaluate the *FlowRank* model, a log of user interactions (e.g., user clicks) with the documents is required. The sampled queries and documents were presented to five evaluators, all graduate students major in Computer Science. For each document, the title and the abstract of the document were displayed to the evaluators. To minimize the potential bias induced by the original document ranking, the search results of each query were shuffled so that the evaluators were not aware of the original ranking produced by the search engine, and the evaluators

[1] http://citeseer.ist.psu.edu

were explicitly told about this process. The evaluators were asked to independently choose one of the three ratings for each document based on their subjective assessment of how relevant a document is to the corresponding query, assuming they had issued the query. The three ratings were:

"Definitely clicked": The evaluator believes he/she should click on the document.

"Probably clicked": The evaluator may or may not click on the document.

"Never clicked": The evaluator believes he/she would not click on the document.

Without loss of generality, each click record consists of a 4-tuple (u, q, d, p_c), where u denotes the user, q denotes a query issued by u, d denote one of the search results, and p_c denotes the probability of the user u clicking on the document d. Given the editorial ratings by the evaluators, a rating *"Definitely clicked"* was translated into $p_c = 1.0$, *"Probably clicked"* was translated into $p_c = R$ where $R \in (0, 1)$ is a random variable, and *"Never clicked"* was translated into $p_c = 0.0$. Thus we used the obtained data $Dataset_s$ to simulate user clicks.

Relevance judgment for $Dataset_s$ were generated by presenting the same queries and documents to another two evaluators who were both computer scientists. This time, the evaluators were allowed to read the detailed content of the document whenever necessary. Each of the evaluators was asked to rate the document in a five-point scale from 0 to 4, defined as:

0 - Irrelevant match The document did not contain any information about the query.

1 - Marginally relevant match The query terms might appear in the document, but the document was mainly about something else.

2 - Borderline match The document could be rated as 1 or 3.

3 - Fairly relevant match The document contained relevant information about the query terms, however there are more relevant document(s) in the search results.

4 - Best match The document contained highly relevant information about the query terms, and should be selected as the most relevant document among all the search results.

Data collected in this phase is labeled as $Dataset_e$.

A number of metrics have been proposed to quantitatively describe the similarity between queries [8, 25, 27, 32]. We defined the query similarity metric based on the common query title measurement [25] as

$$sim(\mathbf{q_i}, \mathbf{q_j}) = \frac{|T_i \bigcap T_j|}{max(|T_i|, |T_j|)} \tag{3.3}$$

where $\mathbf{q_i}$ and $\mathbf{q_j}$ are were two queries, T_i and T_j were the terms in the titles of the documents returned by the search engine. In other words, the similarity between the two queries was in proportion to the number of common terms in the titles of the search results. If the titles were similar, the queries were also similar.

User similarity metrics are commonly used in collaborative recommendation and filtering systems, because by learning from other *"like-minded"* users, the system can predict a user's preference [25]. In the *FlowRank* model, user similarity is implicitly expressed by the arc capacities because of step (2) of the graph transformation algorithm in Section 4.3.1. And the user similarity becomes the upper-bound for the flow values. For simplicity of computation, we currently treat all users the same, so for any two users $\mathbf{u_i}$ and $\mathbf{u_j}$, $sim(\mathbf{u_i}, \mathbf{u_j}) = 1$. We justify this assumption by noting that all of the five evaluators were graduate students in Computer Science and most likely familiar with the search topics. However, it would be interesting to observe the impact of different user similarity metrics in the next phase of evaluation.

Furthermore, we obtained the relevance ranking scores computed by the search engine for all the (query, document) pairs.

Finally, we were able to build a transformed search graph for each of the sampled queries, and applied the *FlowRank* algorithm to compute the network flows and generate the collaborative rankings. We used adjacent matrices to represent the transformed search graphs, with elements being the corresponding arcs' capacities. We computed the values of the network flows based on Rothberg's implementation of Goldberg's Push-Relabel algorithm [50], which is usually considered

the fastest in practice.

3.4.2 Runtime Performance

We first measured the runtime performance of *FlowRank* in order to get an estimation of online execution efficiency. 10 batch runs of *FlowRank* were performed on the D_s dataset on a workstation equipped with an Intel Xeon 2.8GHz CPU and 1G RAM running Fedora Core 3. The elapsed real time t between the invocation and termination of the ranking module was measured using the system standard *time* command. By averaging the 10% trimmed-mean of t among the 100 queries, we estimated that it took approximately 0.109 seconds to compute the document rankings for each query, with 0.0218 seconds contributed by the graph transformation process and the remaining 0.0872 seconds by the network flow calculation. For a much larger dataset, the vast majority of the runtime will be contributed by the network demands of retrieving information about the search results, the queries, and the users in the process of building and transforming the graph. One solution is to pre-compute and cache locally (part of) the graph.

3.4.3 Results and Discussion

The ranking computed by *FlowRank* (F) was compared with those generated by two baseline ranking algorithms: the HITS algorithm [71] (H) and the original ranking computed by *CiteSeer* (S). We used the Discounted Cumulative Gain (DCG) metrics [58, 60] to measure the quality of the relevance ranking. The major motivation to use DCG is that it assigns heavier weights to higher-ranked documents, and allows us to have different levels of subjective relevance judgment by the evaluators. Given a query $\mathbf{q}$, DCG is defined as

$$DCG(\mathbf{q}) = \sum_{d=1}^{N} \frac{2^{R(d)} - 1}{\ln(1+d)} \tag{3.4}$$

where $R(d)$ is the editorial rating of the d-th webpage in the top N search results. Here the ratings from the two evaluators in D_e were averaged as the editorial ratings $R(d)$ to be used in the DCG calculation.

DCG can be represented in the form of a vector $DCG(\mathbf{q}) =< r_1, r_2, r_3, ..., r_N >$,

where r_i represents the DCG at rank position i. On the other hand, for each query, an ideal (a.k.a. optimal) gain vector I can be arbitrarily determined and plugged into Equation 3.3 to obtain an ideal DCG vector $DCG_I =< b_1, b_2, b_3, ..., b_N >$. In our evaluation, the ideal gain vector I and ideal DCG vector DCG_I were defined as

$$\begin{aligned} I &= < 4, 3, 3, 2, 2, 2, 1, 1, 1, 1, 0, 0, ... >, \\ DCG_I &= < 21.64, 28.01, 33.06, 34.92, 36.59, 38.13, 38.61, 39.07, 39.50, 39.92, 39.92, 39.92, ... > . \end{aligned}$$

Intuitively, a higher DCG score indicates a better relevance ranking of the search results. DCG was computed for the top 20 ranked documents in H, S, and F, since previous study showed that users seldom looked beyond the first few result pages [57].

Compared with the two baseline algorithms, *FlowRank* achieved substantial improvements on the relevance rankings. This advantage was reflected in the average DCG metrics. In both comparisons, *FlowRank* was able to improve the relevance ranking for about 47% of the queries. Across all ranking positions, the average DCG for F was 59.42, compared with 57.54 for S and 57.37 for H. The number of queries with DCG increased, decreased, and without change was summarized in Figure 3.9 (a).

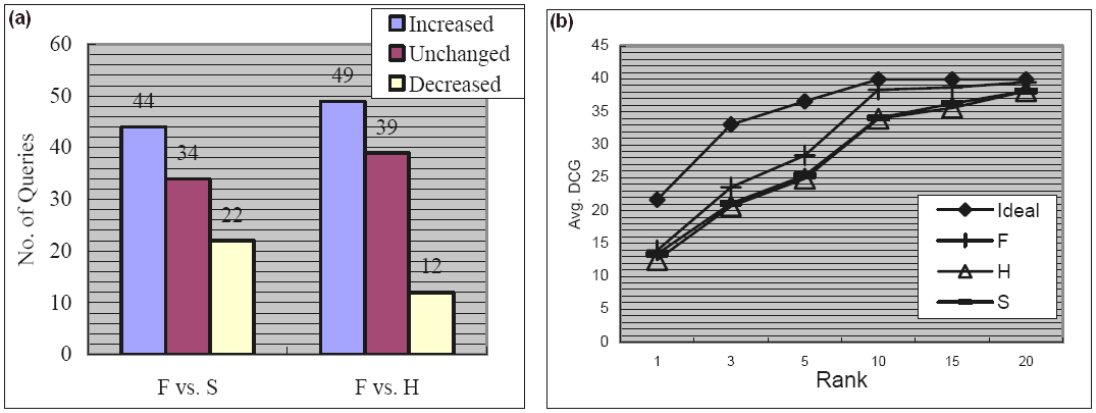

Figure 3.9. (a) DCG changes for queries; *FlowRank* was able to improve the relevance ranking for about 47% of the queries. (b) Average DCG curves; *FlowRank* (F) outperformed both baseline algorithms H and S by about 15%.

Figure 3.9 (b) plots the average DCG curves for the three ranking algorithms (F, H, and S) together with the ideal DCG (DCG_I) at ranking positions 1 to 20. We see that *FlowRank* outperformed the other two baseline algorithms and quickly approached the ideal curve, which began to level off at the rank 10. This was explained by the fact that the documents ranked below position 10 were indeed less relevant.

Does collaborative ranking with *FlowRank* really help? We further investigated whether there is indeed a correlation between the collaborative contribution exploited by *FlowRank*, and the amount of increase in the average DCG over the two baseline algorithms. We cross-examined the improvement in the DCG score for each of the queries with the size of the transformed collaborative search network. Here we made an assumption that the more users have involved in and contributed to the ranking process, the larger the collaborative search network will be. A correlation test was performed on two variables: size of the collaborative search network S_g and increase in the DCG IG. We found that the Pearson correlation of S_g and IG was 0.334 with a P-value of 0.033 (α-level was 0.05), confirming a positive correlation between the two.

3.4.4 Remarks

We simplified in the current implementation of the flow-based model, in which we assigned the same capacities to the arcs between u'_i and $\mathbf{u^t}$, indicating that all collaborators were identical. We also performed three batch runs of the *FlowRank* algorithm, during which the number of users was changed on purpose. Although we observed the correlated changes to the DCG improvement, the correlation was not statistically significant. We believe this was mostly due to the small number of collaborators.

We calculated the maximum flow values using the Push-Relabel algorithm [50], because the sizes of the graphs were relatively small, and runtime efficiency was a primary concern. However, depending on the scale (and often the subject domain) of a collaborative search, the graph can become very dense or sparse, or can become so huge that in-memory access to the entire graph is impractical. One solution is to use a different maximum-flow calculation algorithm, e.g., the shortest

augmentation path algorithm [39].

Finally, we discuss the alternative definitions of the arc capacities in *FlowRank*. One advantage of *FlowRank* is its flexibility in allowing such alternative definitions. Under the network flow framework, we can derive the arc capacities based on the external or internal, explicit or implicit feedback among the users, queries, and pages. For example, in the aforementioned instantiation of this framework, we leverage the common query title measurement [25] as the query similarity metric, and base the capacity definitions on the implicit feedback of user clicks. However, we can also derive the arc capacities based on additional information sources other than the search logs, e.g., the similarity between two user profiles, and the similarity of two pages based on the $tf \cdot idf$ metric. An assumption is to assign sufficiently large capacities to the arcs between $\mathbf{q^t}$ and $\mathbf{d_i^t} \in \mathbb{D}_t$, so that the network flows are not bound by $C(\mathbf{q^t}, \mathbf{d_i^t})$. If this assumption is satisfied, we can apply alternative definitions of the arc capacities and the network flow-based framework will still be able rank the documents accordingly.

3.5 Ranking as Graph Partitioning

In this section, we further exploit the *FlowRank* model by considering ranking as a flow-based graph partitioning problem. Previously, Flake et al. have proposed to use flow-based metrics to identify online communities that are internally highly cohesive [41, 42]. Intuitively, locating these communities is a problem similar to that of finding sub-graphs that are separable from the rest of the universal Web graph.

Applying the *FlowRank* model onto Web search, ranking the retrieved documents can also be considered as a graph partitioning problem, briefly described as follows.

Assume that the definitions in the *FlowRank* model are still followed. Let G denote the whole Web graph in which the vertices $d \in D$ represent the Web pages. Let S denote a sub-graph of G, such that for all vertices $\mathbf{d} \in S$, $\mathbf{d}$ has a flow value that is larger than that of any vertex $\mathbf{d'} \in (G - S)$. By partitioning G into two sub-graph S and $(G - S)$, we can obtain in S for query $\mathbf{q^t}$ the top-$|S|$ ranked documents. Figure 3.10 sketches an intuitive depiction of the graph partition.

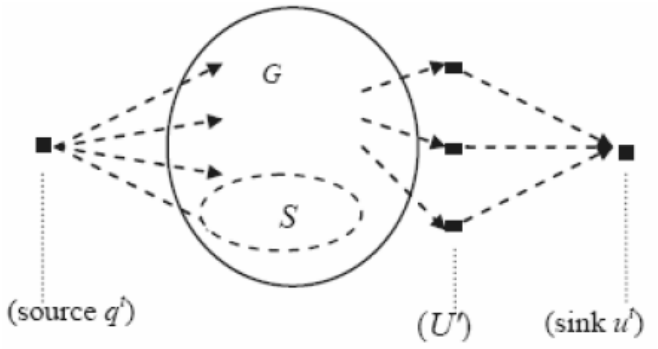

Figure 3.10. Ranking as graph partitioning: find the top N ranked Documents in sub-graph S of G according to the flow values from $\mathbf{q^t}$ to S and $(G - S)$.

3.6 Summary

We have proposed a flow-based graph model of collaborative ranking which exploit user feedback on the relationships among similar queries and relevant documents, and the users' collective access pattern on the retrieved documents. Our model translates the collaborative ranking problem into a flow-calculation problem in a collaborative search network. This unique perspective casts the ranking problem as a network flow problem for which there is a large body of work. Derived from the model, we present a collaborative ranking algorithm based on the network flows. Evaluations in the document search domain shows our algorithm effectively improves the relevance ranking computed by two baseline algorithms. In Appendix, we apply our model to further discuss the implications of several representative collaborative ranking scenarios.

Chapter 4

Re-ranking Search Results Based on User Feedback

4.1 Overview

In the previous chapter, we describe a flow-based model to leverage the collective feedback in a collaborative search process to improve the quality of relevance ranking. Query logs of large-scale search engines contain the queries issued by a huge number of users. They are the sources of collective implicit feedback on users' *search intents*, i.e., *"what typical users are looking for"* with a search query. In this chapter, we propose a relevance ranking algorithm, *Q-Rank*, to leverage the implicit feedback from the logs about the users' search intents and apply such knowledge to effectively refine the relevance ranking of Web search results. *Q-Rank* is applicable to general purpose as well as vertical search ranking systems.

Recall that we denote in Equation 3.1 of Chapter 3 the similarity of the target query $\mathbf{q^t}$ and an alternative query $\mathbf{q_j}$ in the query logs as $sim(\mathbf{q_j}, \mathbf{q^t})$, and in Equation 3.3 we define $sim(\mathbf{q_j}, \mathbf{q^t})$ based on the common query title measurement [25].

Let $\mathbb{Q}_{\mathbf{q^t}} = \{\mathbf{q'}|sim'(\mathbf{q'}, \mathbf{q^t}) \geq \delta\}$ denote the set of query reformulations of $\mathbf{q^t}$ in the search query log, where $sim'(\cdot)$ is a different criterion function. In this chapter, we describe a similarity criterion function $sim'(\cdot)$ such that we define $\mathbb{Q}_{\mathbf{q^t}}$ as the *query context* (see Section 4.3 for details), and present a method *Q-Rank* to use

the query context to improve the relevance ranking of the search results.

The rest of this chapter is organized as follows. We first describe the motivation, algorithm design, and implementation of *Q-Rank*. Then we discuss, in detail, the experiments for fine-tuning the algorithm parameters as well as the evaluation results. We also discuss several interesting scenarios in the framework for employing query log data in ranking, and finally summarize our study.

4.2 Motivation

First, let us consider the following two search scenarios:

Naive queries. The query is an unarticulated query consisting of *naive* search terms. For example, a user issues the query *"hard disk case"* despite the fact that a more accurate description of the user's intent (i.e., generally accepted and more frequently used on the web) is *"hard drive enclosure"*. Because search engines usually rank the webpages based on their syntactic match with the query terms (i.e., considering the term frequency, proximity, etc.), the search results for this query could suffer in terms of relevance, even though some of the retrieved pages may actually contain the more accurate descriptive terms (such as *"drive"* and *"enclosure"*).

Recency queries. A user wants to find information about the upcoming lunar eclipse by issuing a query *"lunar eclipse"*. Although we hypothesize that recently a considerable amount of queries containing both the phrases *"lunar eclipse"* and *"2008"* could have been repeatedly sent to popular search engines, these search engines may still not be able to rank the official 2008 lunar eclipse website as the top result for the query *"lunar eclipse"* without specifying the year *"2008"*. For recency queries, more up-to-date information is *implicitly* favored.

Both the above scenarios present a challenge to the search engine, and call for relevance ranking methods that take into account not only a webpage's overall quality and relevance to the search query, but also the match with the users' informational need, further referred to as their real *search intents*.

Q-Rank is based on an intuitive rationale, that the most frequently repeated *query extensions* of a target query and the *adjacent queries* provide important information about users' search intents. Here *query extensions* mean the terms extracted from queries that contain the target query as an affix, and *adjacent queries* mean the queries that immediately precede or follow a particular user query in the same search session. For example, given the target query "aquarium", the most frequent extensions, as observed in a real search engine query log, are *"fish", "supplies", "screensaver", "tanks"*, and *"lighting"*, the queries that most frequently follow it are *"aquarium screen saver", "aquariums", "aquarium supplies", "fish tanks", "aquarium fish", "aquarium screensaver", "aquarium filters"*, and *"tropical fish"*, and the most frequent queries that precede it (ignoring misspellings) are *"aquariums", "fish", "fish tank", "zoo", "petco", "aquariophilie", "aquarium screensaver"*, and *"marine aquarium"*.

Formal definitions of query extensions and adjacent queries will be given in the next section. Intuitively, the distribution over query extensions and adjacent queries in the search engine query logs at any point in time can be treated as snapshot of the typical user interests related to the concept in a target query; thus, when a user submits the target query, it can be assumed that she/he may be interested in a collection of documents that closely match this distribution. Previous studies [72, 78] have confirmed that when a user is not satisfied with the current set of search results, the user is very likely to refine or rewrite the query for the purpose of generalization, specialization, or adding new information, as illustrated in Scenario A in Figure 4.1. The correlation between a query and another one that frequently follows it in user search sessions can thus be regarded as an important clue about how users try to adjust their lexical choices to better match their search intents to the Web content and the Web search process. Hence, queries that either frequently follow or frequently precede a target query may contain useful lexical information about documents that would satisfy the user's search intents (the former capture better the attempt to disambiguate or match the Web content, and the latter capture better the original user intent, especially when the users subtract terms from their original queries).

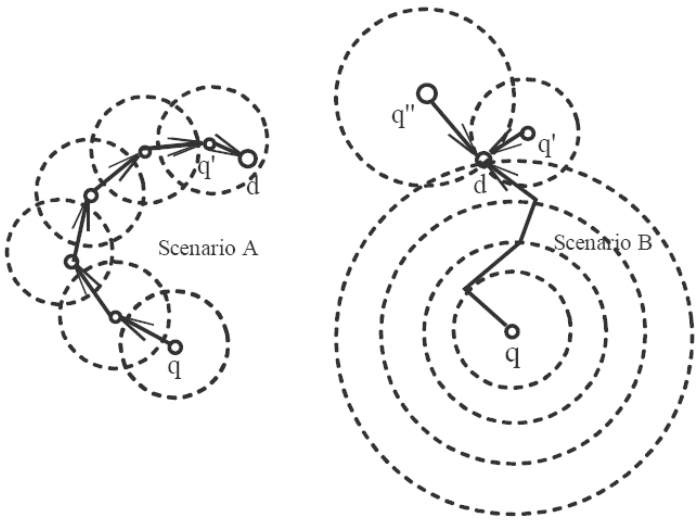

Figure 4.1. User behavior in the search space, in which a target query **q** is followed by a sequence of actions that eventually lead to a relevant document **d**. Scenario A: the user continuously reformulates the query until reaching a relevant document. Such reformulations (made by previous users) are exploited by *Q-Rank* to better compute the match between documents retrieved by a search engine for the target query **q** and the typical intent(s) of the users who have issued **q**. Scenario B: circles represent different tiers in the ranked list; the user keeps on browsing the search result pages until finding a relevant document. This document may have been retrieved by other queries somewhat related to the initial query and/or contained terms from these queries.

4.3 Algorithm Design

In this section, we define the concept of the query context and introduce the *Q-Rank* algorithm.

4.3.1 Query Context Definition

We formalize the definition of *query context* as follows. Let $\mathbb{Q}_t$ denote the set of queries in a search engine query log for a given time window t. We use the notation $\mathbb{Q}_+(\mathbf{q^t})$ for the set of queries which were seen following a query $\mathbf{q^t}$ in the user search sessions, and $\mathbb{Q}_-(\mathbf{q^t})$ for queries which were seen preceding it. The union of $\mathbb{Q}_+(\mathbf{q^t})$ and $\mathbb{Q}_-(\mathbf{q^t})$ will be referred to as the adjacent queries $\mathbb{Q}_{adj}(\mathbf{q^t})$:

$$\mathbb{Q}_{adj}(\mathbf{q^t} \doteq \{\mathbf{q}_{adj} | \mathbf{q}_{adj} \in \mathbb{Q}_+(\mathbf{q^t}) \cup \mathbb{Q}_-(\mathbf{q^t})\} \tag{4.1}$$

For example, for the query *"aquarium"*, the adjacent queries include *"fish tanks"*, *"aquarium fish"*, *"aquarium screensaver"*, *"aquariums"*, *"fish"*, *"fish tank"*, *"zoo"*, and *"petco"*, etc. that appear either before or after the query *"aquarium"* in search sessions.

We define the *user-based expansions* of a query $\mathbf{q^t}$ as those queries that contain $\mathbf{q^t}$ as an affix and are logged in a time window t. We employ only expansions in which $\mathbf{q^t}$ is a prefix for efficiency consideration, and we further define the query extensions $\mathbb{Q}_{ext}$ of a given query $\mathbf{q^t}$ as being all such expansions from which the common prefix $\mathbf{q^t}$ is removed. Formally,

$$\mathbb{Q}_{ext}(\mathbf{q^t}) \doteq \{\mathbf{q}_{ext} | \mathbf{q^t} \cdot \oslash \cdot \mathbf{q}_{ext} \in \mathbb{Q}_t\}, \tag{4.2}$$

where $\oslash$ is a term delimiter (e.g., an empty space character) and $\cdot$ denotes the operation of string concatenation.

For example, for the previous query *"aquarium"*, the query extensions include *"stands"*, *"plants"*, *"supply"*, and *"filters"*, etc.

When computing $\mathbb{Q}_{ext}(\mathbf{q^t})$ and $\mathbb{Q}_{adj}(\mathbf{q^t})$, we employ some preprocessing steps to normalize the queries and aggregate statistics over near duplicates: basic stemming, punctuation removal, word order normalization, and spell checking [30].

In practice, when $\mathbb{Q}_{ext}(\mathbf{q^t})$ is empty, we use instead $\mathbb{Q}_{ext}(\overline{\mathbf{q^t}})$, where $\overline{\mathbf{q^t}}$ is the longest prefix query of $\mathbf{q^t}$ for which there exist two or more query extensions, but not more than a predefined small number ξ of extensions (in this way, backing-off from queries such as *"john malkovich"* to very general prefixes such as *"john"* is avoided). Considering again the example of search query *"hard disk case"*, we retrieve adjacent queries such as *"portable hard disk"* and *"usb external hard drive"* from the same user search sessions. By backing off to the query *"hard disk"* because of the lack of extensions for the target query, we identify the popular extensions *"drive"*, *"data recovery"*, *"repair"*, *"utilities"*, *"failure"*, *"problems"*, *"eraser"*, and *"enclosure"*, etc. The words in these adjacent queries and extensions will be used to re-rank the search results retrieved for the original query, under the assumption that compared with the original underspecified query, this additional information can help to better represent the typical user's real search intent.

Finally, we define the query context of a query $\mathbf{q^t}$ as $\mathbb{Q}_{\mathbf{q^t}} = \mathbb{Q}_{ext}(\mathbf{q^t}) \bigcup \mathbb{Q}_{adj}(\mathbf{q^t})$.

In the rest of this chapter we will omit the argument $\mathbf{q^t}$ from the notations

wherever there is no ambiguity in doing so.

4.3.2 Calculating the Re-ranking Scores

Let $\mathbf{q^t}$ denote the original query issued by a user, referred to as the target query, and $\mathbb{D}_{\mathbf{q^t}}$ (or simply $\mathbb{D}$ when no ambiguity arises from omission) denote a set of candidate documents for ranking. While in this work $\mathbb{D}$ contains the top-n ranked documents returned by a search engine for the target query, there can be alternative definitions for the candidate set $\mathbb{D}$. We assign a ranking score for each document $\mathbf{d} \in \mathbb{D}$ based on its lexical overlap with a set of most popular query extensions and adjacent queries to $\mathbf{q^t}$, as in Equation 4.3. The numerator of this formula has two terms, which correspond to query extensions and adjacent queries, respectively. The $tf \cdot idf$ scores [108] are assigned to each query-document pair. They are then weighted by the natural logarithm of the normalized query log frequency in order to account for the difference in query popularity. Each of the terms is weighted by the dampening factor, summed, and then divided by the initial rank of the document. This latter step is done to account for the initial ranking calculated by the search engine, which makes use of many features that are not available at re-ranking time, such as the static rank of a document, the anchor text of links that point to the document, etc.

$$RS(\mathbf{d}, \mathbf{q^t}) \doteq \gamma \cdot \frac{\sum_{i=1}^{|\mathbb{Q}_{ext}|} tf(\mathbf{q_i}, \mathbf{d}) \cdot \ln \frac{|\mathbb{D}|}{|\mathbb{D}_{\mathbf{q_i}}|} \cdot \ln \frac{qf(\mathbf{q_i})}{\sum_{j=1}^{|\mathbb{Q}_{ext}|} qf(\mathbf{q_j})}}{R(\mathbf{d})} + (1-\gamma) \cdot \frac{\sum_{i=1}^{|\mathbb{Q}_{adj}|} tf(\mathbf{q_i}, \mathbf{d}) \cdot \ln \frac{|\mathbb{D}|}{|\mathbb{D}_{\mathbf{q_i}}|} \cdot \ln \frac{qf(\mathbf{q_i})}{\sum_{j=1}^{|\mathbb{Q}_{adj}|} qf(\mathbf{q_j})}}{R(\mathbf{d})} \quad (4.3)$$

Equation 4.3. *Q-Rank* document scoring function. $\mathbb{Q}_{ext}$ and $\mathbb{Q}_{adj}$ denote the query context sets: extensions and adjacent queries to $\mathbf{q^t}$, respectively. $\gamma \in [0...1]$ denotes a dampen factor leveraging the contribution of each type of query context. $tf(\mathbf{q_i}, \mathbf{d})$ denotes the frequency of the query context $\mathbf{q_j}$ in document $\mathbf{d}$. $\mathbb{D}_{\mathbf{q_i}}$ contains all documents $\mathbf{d}$ for which $tf(\mathbf{q_i}, \mathbf{d}) > 0$. $qf(\mathbf{q_j})$ denotes the logged frequency of query $\mathbf{q_j}$. $R(\mathbf{d})$ denotes the initial rank of $\mathbf{d}$.

The impact of biasing the re-ranking process towards the initial ranking of the search engine is evaluated in the development experiments in the next section. As expected (because of the extra ranking features used by the search engines), we find out that *Q-Rank* performs better with such a bias.

4.3.3 The *Q-Rank* Algorithm

Algorithm 2 The *Q-Rank* algorithm for re-ranking search results.

Require: The initial query: $\mathbf{q^t}$,
Queries in the query log within a time=t window: $\mathbb{Q}_t$,
Query context composition parameter: op,
The number of document candidates to re-rank: c,
Output ranking range: n,
The number of unchanged top-ranked documents: u,
A set of initially retrieved documents: $\mathbb{D} = \{\mathbf{d_1}, \ldots, \mathbf{d_{c'}}\}$ retrieved for $\mathbf{q^t}$, where $c' \leq c$.
Ensure: Re-ranked document set $(\mathbf{d}_{\pi(\mathbf{1})}, \ldots, \mathbf{d}_{\pi(\mathbf{c'})})$, where π is a permutation of $1, ..., c'$.
Construct $\mathbb{Q}_{ext}(\mathbf{q^t})$ and $\mathbb{Q}_{adj}(\mathbf{q^t})$ from $\mathbb{Q}_t$, given $\mathbf{q^t}$ and op.
for all document $\mathbf{d} \in \mathbb{D}$ **do**
  $score_d = RS(\mathbf{d}, \mathbf{q^t})$
end for
if u = 0 **then**
  Sort in descending order the documents $\mathbf{d}$ by $score_d$
else
  Sort the documents $\mathbf{d_{u+1}}, \mathbf{d_{u+2}}, \ldots, \mathbf{d_{c'}}$ in descending order of $score_d$.
end if
return $(\mathbf{d}_{\pi(\mathbf{1})}, \ldots, \mathbf{d}_{\pi(\mathbf{c'})})$

In the *Q-Rank* algorithm, *op* specifies the composition of the query context $\mathbb{Q}_{\mathbf{q^t}}$ (i.e., whether $\mathbb{Q}_{ext}$ and/or $\mathbb{Q}_{adj}$ are used to construct the query context), and the size of these sets ($|\mathbb{Q}_{ext}|$ and $|\mathbb{Q}_{adj}|$)); c specifies how many document candidates to consider and n specifies the output ranking range (obviously, $n \leq |\mathbb{D}| \leq c$); u is the number of top-ranked search results of which the ranking is unchanged; i.e., if u is larger than zero, only the bottom $n - u$ of the top n results will be re-ranked.

$\mathbb{D}$ can be computed based on the snippets generated by the search engine, or contain the full-text of the retrieved webpages. However, the latter approach

requires access to the text of these pages and considerably much more expensive computation. We discuss its applicability and impacts on relevance ranking in Section 4.4.

Collapsing [38] refers to grouping together webpages from the same domain as a single entry in the search results, and is a common practice for major Web search engines. While it provides navigational context for users, it can also mislead *Q-Rank* to incorrectly interpret the initial ranking of the retrieved results. To minimize such negative impacts, we removed the duplicate documents in the *Q-Rank* experiments. That is, if there are two or more documents from the same top-level domain, only the one with the highest rank is retained. All other duplicates are removed from the candidate documents set $\mathbb{D}$.

4.4 Experimentation

4.4.1 Datasets

To evaluate *Q-Rank*, we were granted access to a data set containing several tens of thousands of queries associated with several million webpages as the search results. Each pair of (query, webpage) was scored by editors on a scale from 0 to 5. The rating reflects the webpage's relevancy to the corresponding search query, 0 being completely irrelevant and 5 being extremely relevant. From this dataset, we randomly sampled two sets of 1,000 queries and the associated search results as our training datasets. Another 2,000 queries were randomly sampled from the remaining data for evaluation. We also had access to a two-month query log of the same search engine, which contained aggregated frequencies of queries and also pairs of queries that were sent in succession by the users.

4.4.2 Evaluation Metrics

The relevance ranking algorithm of a popular commercial Web search engine (MSN Live Search) is used as the baseline, denoted by S. We measure the ranking quality with the Discounted Cumulative Gain (DCG) metric [58, 60]. DCG assigns heavier weight to higher-ranked documents, thus allows us to define various levels of

subjective relevance judgment for the human editors. For a given query $\mathbf{q}$, DCG is defined as

$$DCG(\mathbf{q}) = \sum_{d=1}^{n} \frac{2^{R(d)} - 1}{\ln(1+d)} \tag{4.4}$$

where $R(d)$ is the editorial rating of the d-th webpage. A higher DCG reflects a better ranking of the results. DCG for the top n results generated by *Q-Rank* and S are computed and compared. Because we cannot present the absolute DCG values on this data set, we will quantify the performance gains in relative terms.

4.4.3 Development Experiments

4.4.3.1 Comparison of three sets of parameters

Table 3.1 (a, b, and c) summarizes the three sets of parameter settings investigated: only $\mathbb{Q}_{ext}$ are used, only $\mathbb{Q}_{adj}$ are used, and both query contexts are used. The notation $|\cdot|$ refers to the number of query extensions and adjacent queries employed by *Q-Rank*. When adjacent queries are used, we employ an equal number of preceding and subsequent queries: e.g., $|\mathbb{Q}_{adj}| = 20$ means that 10 preceding and 10 subsequent queries are employed. For each of the three sets of experiments, c denotes the number of document candidates for re-ranking, u denotes that the top-u documents are left untouched, and a Boolean parameter *bias* indicates whether or not to *bias* towards the original ranking (the factor $R(\mathbf{d})^{-1}$ in Equation 4.3). If $bias = false$, we assign $R(\mathbf{d}) = 1, \forall \mathbf{d} \in \mathbb{D}$.

4.4.3.2 Various query lengths

Figure 4.2 (a, b, and c) shows the percentage of queries with improved ranking (measured in term of the DCG metric), broken down by various query lengths for each of the three sets of parameters. When computing DCG, we assign the output ranking range $n = 15$. In all parameter settings except SE_1 and SB_1, *Q-Rank* improves the DCG scores for more than 50% of the queries with changed rankings, which cover between 64% and 72% of all the queries. These are major improvements considering the fact that for each query, on average, less than 20 documents have relevance ratings assigned by the editors (the top 10 search results

Table 4.1. The investigated parameter space of *Q-Rank*.

Parameters	$\lvert Q_{ext}\rvert$	c	u	*bias*
SE_1	20	50	1	*false*
SE_2	20	50	2	*false*
SE_3	20	50	2	*true*

(a) $\mathbb{Q}_{ext}$ only ($\gamma = 1$)

Parameters	$\lvert \mathbb{Q}_{adj}\rvert$	c	u	*bias*
SA_1	20	50	1	*false*
SA_2	20	50	2	*false*
SA_3	20	50	2	*true*

(b) $\mathbb{Q}_{adj}$ only ($\gamma = 0$)

Parameters	$\lvert \mathbb{Q}_{ext}\rvert \bigcup \lvert \mathbb{Q}_{adj}\rvert$	c	u	*bias*
SB_1	20+20	50	1	*false*
SB_2	20+20	50	2	*false*
SB_3	20+20	50	2	*true*

(c) Both $\mathbb{Q}_{ext}$ and $\mathbb{Q}_{adj}$

are usually among those judged), and the default rating for documents without editorial judgment is 0. Consequently, when replacing a document with a low but positive rating with an un-judged but potentially relevant document, the DCG score will be decreased.

The numerical values for this first set of experiments are shown in Figure 4.3 (a) An important observation is that when taking into account the initial ranking (i.e., *bias* = true), *Q-Rank* performs better across all sets of parameters (SE_3, SA_3, and SB_3 consistently outperform SE_1/SE_2, SA_1/SA_2, and SB_1/SB_2). Another observation is that using adjacent queries $\mathbb{Q}_{adj}$ alone seems to produce the best ranking among the three sets; this is further verified in a number of experiments discussed in the following sections.

We also observe a consistent pattern in which keeping the top two search results unchanged ($u = 2$) and biasing towards the original ranking ($bias = true$) outperform the other settings for all query lengths - with one exception (three-

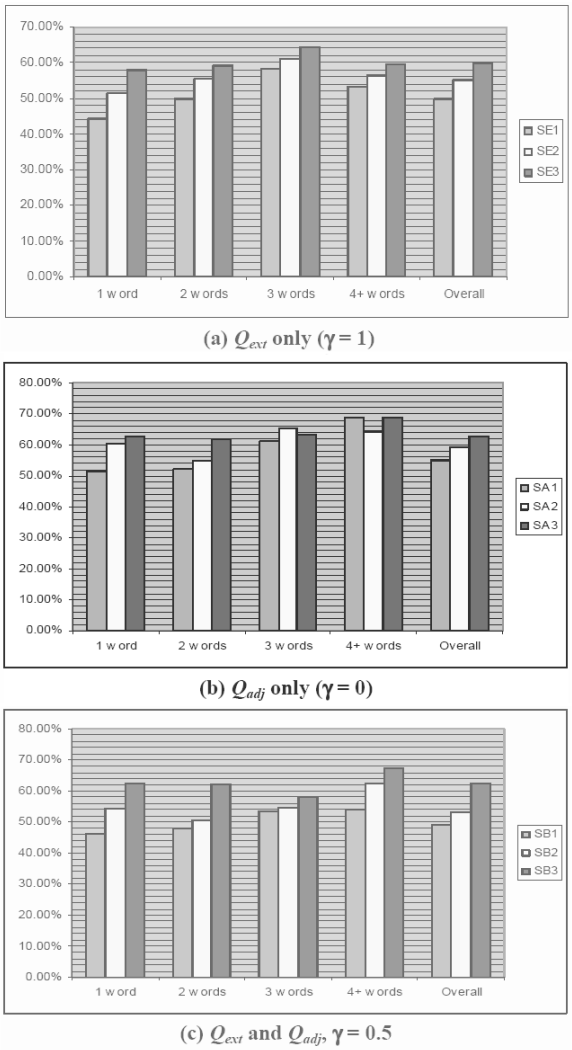

Figure 4.2. Percentage of queries with improved ranking broken down by their length in terms of number of words.

words queries in the SA settings). Although we initially expect that most of the improvements would come from one-word queries, due to the richness of the search query logs and to the back-off strategy employed in generating the query context sets, *Q-Rank* is able to improve the ranking substantially even for longer queries (three or more words). *Q-Rank* performed better with a bias towards the original relevance rankings, for further experiments only the parameter settings SE_3, SA_3, and SB_3 are used.

4.4.3.3 Various ranking ranges

Figure 4.3 (b) reports the performance of *Q-Rank* broken down by various re-ranking ranges (n). Clearly, the increase in n does not guarantee a better performance. In fact, *Q-Rank* produces the best ranking constantly for $n = 10$ across all three sets of parameters. As discussed previously, this can be explained by the fact that most results with lower initial ranks are not judged and have a default editorial rating of 0. Once again we see that when using $\mathbb{Q}_{adj}$ alone *Q-Rank* performs the best (SA_3, $n = 10$). One explanation is that adjacent queries represent the most frequent modifications of the underspecified query when the users are not satisfied with the search results. As a result, such reformulated queries are good representations of the users' real search intents.

4.4.3.4 Various numbers of unchanged top results

Figure 4.3 (c) summarizes the performance of *Q-Rank* broken down by various numbers of unchanged top results (u), in which the top u ranked results remain the same. $SE'_3/SA'_3/SB'_3$ indicates that except for u, other parameters remain the same as in $SE_3/SA_3/SB_3$. Here we assign $n = 10$ because it has been shown to achieve the best performance in the previous experiments. While *Q-Rank* is capable of pushing up relevant documents that had been initially ranked lower, assigning u to be an empirically-determined positive number produces better results. There are several good reasons to fix the top-u results. First, some websites are designed completely in images or Flash animations with few or even no textual contents, e.g., the homepage of the famous German automobile maker, Mercedes[1]. Because it is extremely hard or even impossible to generate meaningful snippets for such websites, *Q-Rank* would not be able to compute a meaningful re-rank score. Second, major commercial search engines often employ a list of definitives to be shown as the top results (i.e., webpages that are editorially matched to queries), especially for well-known organizations and businesses. This also suggests that we should keep the top one or two results unchanged during the re-ranking process.

[1] http://www.mercedes-amg.com/

4.4.3.5 Various numbers of re-rank candidates

Figure 4.3 (d) reports the performance of *Q-Rank* broken down by various numbers of document candidates c considered in the re-ranking process. Here we assign $u = 2$ because it has been shown to achieve the best result with this setting in previous experiments. We observe that increasing the number of re-rank candidates c does not yield better performance, and this observation is consistent across all sets of parameters. A possible explanation is that there is no sufficient annotated data: it is very likely that most of the rated websites are already in the top 30 results, and thus taking into account a larger pool of document candidates does not necessarily improve the DCG score of the final ranking, or may even deteriorate the ranking by replacing the higher-ranked relevant documents with irrelevant ones.

4.4.3.6 Selection of γ

We run a series of experiments on the training datasets to optimize the parameter γ for *Q-Rank*. Figure 4.4 (a) plots the percentage of queries with improved DCG scores as γ increases from 0 to 1 in 0.1 increments. On average, *Q-Rank* improves the relevance ranking for 75.8% of the re-ranked queries. The percentile peaks at $\gamma = 0$ (78.5%), which is consistent with our previous finding that using adjacent queries alone achieves the best results.

Figure 4.4 (b) shows the relative increase (in percentage) of the DCG scores when γ grows from 0 to 1 in 0.1 increments. When we assign $\gamma = 0.5$, the DCG scores are increased by an average 6.81% for 76.31% (538 out of 707) of the re-ranked queries; this amounts to more than half (53.8%) of the queries in the training dataset.

4.4.3.7 Full-text vs. document snippets

To study the performance difference of using full-text webpages and document snippets in *Q-Rank*, we download the documents retrieved by the search engine and updated the contents in D to be the full-texts of these documents instead of the snippets. Table 3.2 summarizes the results.

We plot the percentage of queries with improved ranking and the percentage of relative DCG improvement in Figure 4.5 (a) and (b). Not surprisingly, using

(a) The investigated parameter space (n=15).

	Q_{evt} only			Q_{adj} only			$Q_{evt} + Q_{adj}$		
	SE_1	SE_2	SE_3	SA_1	SA_2	SA_3	SB_1	SB_2	SB_3
All queries (Improved/Total changed)	**49.8%** (319/640)	**55.2%** (354/641)	**59.7%** (384/643)	**54.9%** (386/703)	**59.2%** (416/703)	**62.8%** (443/706)	**49.0%** (351/717)	**53.3%** (383/719)	**62.5%** (451/722)
One-word queries	44.4%	51.6%	57.8%	51.5%	60.4%	62.8%	46.3%	54.5%	62.5%
Two-word queries	49.8%	55.4%	59.3%	52.1%	54.7%	61.6%	47.8%	50.5%	62.1%
Three-word queries	58.3%	60.9%	64.4%	61.2%	65.3%	63.3%	53.3%	54.6%	58.1%
Four+ word queries	53.1%	56.3%	59.4%	68.9%	64.4%	68.9%	54.2%	62.5%	67.4%

(b) Various ranking ranges (n).

	SE_3 (c=50, u=2, bias=true)			SA_3 (c=50, u=2, bias=true)			SB_3 (c=50, u=2, bias=true, γ=0.5)		
	n = 10	n = 15	n = 20	n = 10	n = 15	n = 20	n = 10	n = 15	n = 20
All queries (Improved/Total changed)	**63.8%** (410/643)	**59.7%** (384/643)	**59.2%** (377/637)	**66.4%** (469/706)	**62.8%** (443/706)	**62.2%** (439/706)	**62.6%** (452/722)	**62.5%** (451/722)	**59.0%** (426/722)
One-word queries	60.8%	57.8%	55.6%	66.0%	62.8%	60.6%	60.3%	62.5%	53.9%
Two-word queries	63.6%	59.3%	58.4%	64.6%	61.6%	59.8%	62.3%	62.1%	58.2%
Three-word queries	69.6%	64.4%	66.1%	69.4%	63.3%	64.6%	65.1%	58.1%	63.2%
Four+ word queries	62.5%	59.4%	64.5%	71.1%	68.9%	77.8%	66.7%	67.4%	72.9%

(c) Various numbers of unchanged top results (u).

	SE_3' (c=50, n=10, bias=true)			SA_3' (c=50, n=10, bias=true)			SB_3' (c=50, n=10, bias=true, γ=0.5)		
	u = 0	u = 1	u = 2	u = 0	u = 1	U = 2	u = 0	u = 1	u = 2
All queries (Improved/Total changed)	**53.0%** (339/640)	**62.1%** (398/641)	**63.8%** (410/643)	**50.4%** (355/704)	**62.6%** (441/704)	**62.6%** (469/706)	**52.8%** (380/720)	**61.1%** (440/720)	**62.6%** (452/722)
One-word queries	46.7%	60.6e%	60.8%	41.6%	58.4%	66.0%	47.3%	58.6%	60.3%
Two-word queries	55.7%	61.5%	63.6%	51.6%	61.6%	64.6%	56.2%	61.2%	62.3%
Three-word queries	55.7%	64.4%	69.6%	57.8%	67.4%	69.4%	52.6%	61.8%	65.1%
Four+ word queries	56.3%	68.8%	62.5%	57.8%	73.3%	71.1%	54.2%	68.8%	66.7%

(d) Various numbers of re-rank candidates (c).

	SE_3" (n=10, u=2, bias=true)			SA_3" (n=10, u=2, bias=true)			SB_3" (n=10, u=2, bias=true, γ=0.5)		
	c = 20	c = 30	c = 40	c = 20	c = 30	c = 40	c = 20	c = 30	c = 40
All queries (Improved/Total changed)	**75.3%** (490/651)	**78.2%** (507/648)	**73.0%** (473/648)	77.7% (543/699)	**80.3%** (557/694)	**75.1%** (521/694)	**76.3%** (545/714)	**79.3%** (562/709)	**71.0%** (503/709)
One-word queries	67.9%	78.2%	67.3%	74.7%	81.0%	68.1%	73.4%	80.2%	66.8%
Two-word queries	78.8%	77.5%	75.2%	76.5%	78.5%	77.3%	75.6%	77.6%	70.9%
Three-word queries	75.3%	80.4%	76.1%	85.5%	80.6%	76.9%	78.6%	80.0%	73.9%
Four+ word queries	87.2%	79.5%	79.5%	81.6%	87.8%	87.8%	88.2%	84.3%	82.4%

Figure 4.3. Comparison of results for various parameter settings. Percentage of queries with improved ranking is shown.

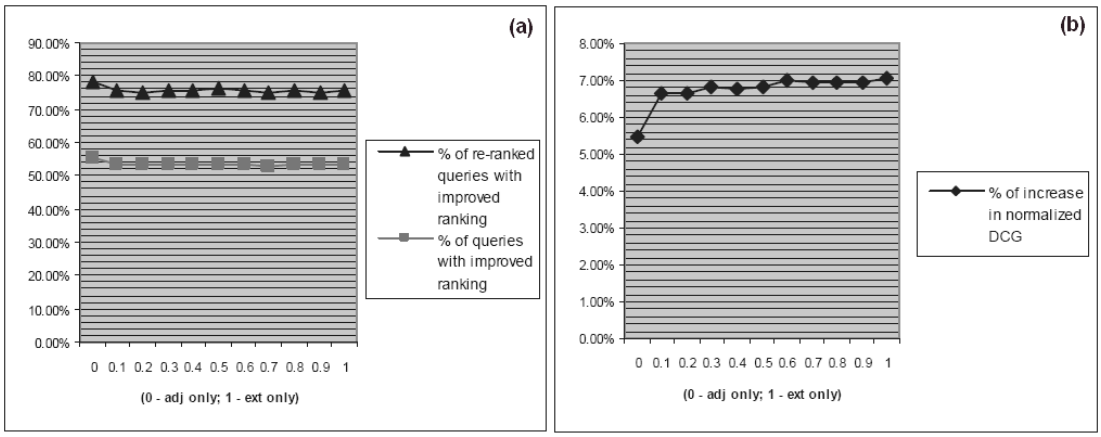

Figure 4.4. Development experiments on selecting γ. (a) Percentage of queries with increased DCG scores when γ grows from 0 to 1 at a step of 0.1. Using adjacent queries alone (i.e., $\gamma = 0$) produces the highest percentage, which is consistent with earlier observations. (b) Percentage of the increased normalized DCG scores when γ grows from 0 to 1 at a step of 0.1.

Queries	$n = 10, u = 2, c = 30, \gamma = 1$	$n = 10, u = 2, c = 30, \gamma = 0$	$n = 10, u = 2, c = 30, \gamma = 0.5$
Improved/Total changed	80.7% (522/647)	81.8% (568/694)	81.7% (579/709)
DCG Improvement	8.7%	8.0%	9.3%

Table 4.2. *Q-Rank* using full-text. Percentage of queries with improved ranking is shown in the first row. Percentage of DCG improvement (*Q-Rank* vs. S) is shown in the last row.

full-text *Q-Rank* has achieved the best performance so far for all three sets of parameters. On average, *Q-Rank* is able to improve the ranking for 81.4% of the queries with an 8.65% increase on average in DCG scores. Once again, we see that using $\mathbb{Q}_{adj}$ alone achieves the largest improvement for the highest percentage of queries, while in practice $\gamma = 0.5$ is a good trade-off.

4.4.4 Evaluation Results

Finally, we test *Q-Rank* on the evaluation dataset which contains 2,000 queries randomly sampled from the query logs of a popular commercial search engine, using the set of parameters that has achieved the best performance on the training dataset. Results are reported in Figure 4.6 (a) and (b), showing an improvement

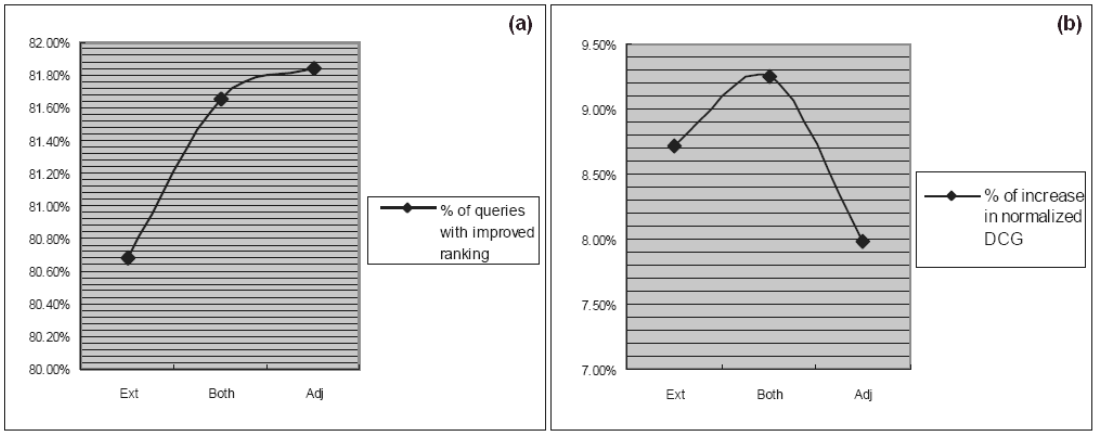

Figure 4.5. Development experiments on full-text. (a) Percentage of queries with increased DCG scores at $\gamma = 1$, $\gamma = 0.5$, and $\gamma = 0$, using full-text of the retrieved documents. Again, using adjacent queries alone ($\gamma = 0$) achieves the best performance. (b) Percentage of increased normalized DCG scores for $\gamma = \{1, 0.5, 0\}$ when using the full-text of the documents.

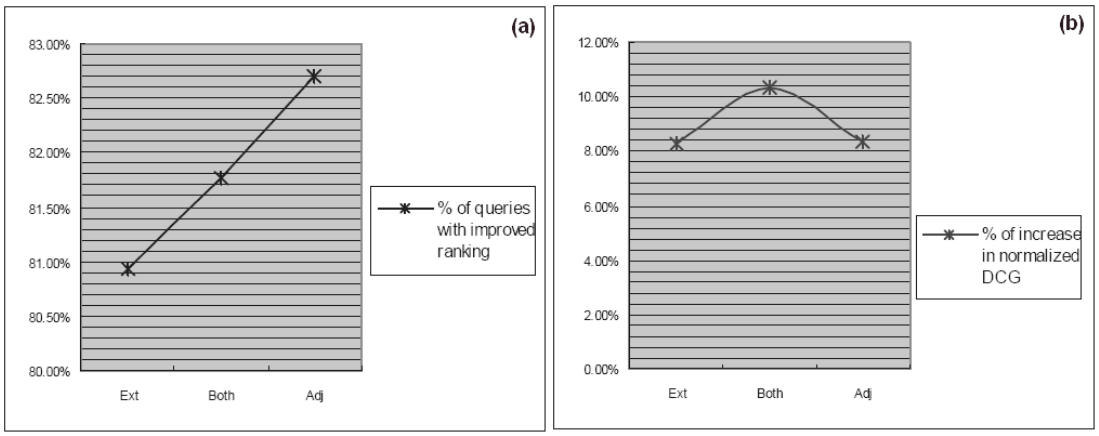

Figure 4.6. Results from test experiments. (a) Percentage of queries with increased DCG scores at $\gamma = 1$, $\gamma = 0.5$, and $\gamma = 0$, using document snippets, on the test set. (b) Percentage of increased normalized DCG scores at $\gamma = 1$, $\gamma = 0.5$, and $\gamma = 0$ on the test set.

on 81.8% of the re-ranked queries with an 8.99% increase in the DCG scores on average. The characteristics of the results are consistent with earlier observations from the training dataset.

Figure 4.7 plots the percentage of queries with improved rankings for various query lengths. We see that there is no consistent pattern in the relevance ranking performance changes that correlates with query length. Note that for longer queries

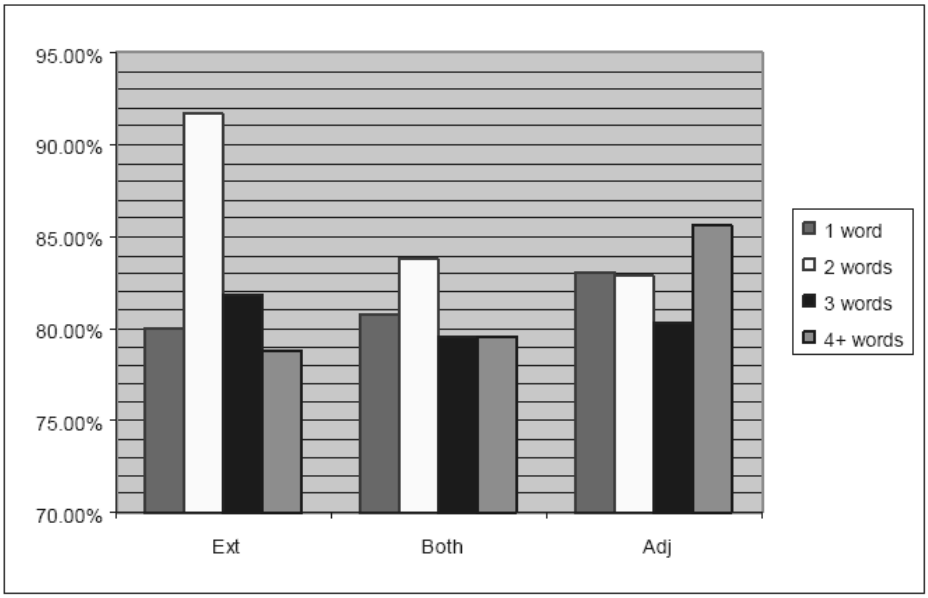

Figure 4.7. Percentage of queries with increased DCG scores in the test set, broken down by query length.

(four words or more), using the adjacent queries seems to perform better. This can be explained on one hand, by the lower number of query extensions and thus the need to back-off, and on the other hand, by the less noisy adjacent query contexts.

4.5 Discussions

Several studies on identifying and studying users' search goals showed that most queries can be categorized as either informational or navigational [67, 106]. Users submit informational queries when they intend to obtain relevant information from the Web and navigational queries when they intend to reach a specific (and typically authoritative) website. The distribution of clicks on the search results of the navigational queries tends to be skewed because the users are often able to recognize the particular website they had in mind [82]. For similar reasons, such queries also have fewer adjacent queries. Thus, Q-Rank's performance with $\mathbb{Q}_{ext}$ and $\mathbb{Q}_{adj}$ suggests that the proposed system works better for informational queries.

Another interesting observation is the variety in the top search results introduced by *Q-Rank*: is it truly beneficial to the search engine users? While query logs contain aggregated knowledge about collective preferences, each user may have

his or her preference as for what should rank higher. For example, Google's top 10 results for the query *"cats"* are mostly about cat, the animal. On the contrary, the top 10 results from *Q-Rank* also include websites about the popular Broadway musical and the Charlotte Area Transit System (CATS). Apparently, the results produced by *Q-Rank* are more diversified, and potentially capture a wider set of interests for a particular query. Such advantage may not be obvious when the results are evaluated with the DCG metrics, because different topics for the same query are not distinguished as long as they are all relevant. On the other hand, an interesting modification to *Q-Rank* is to consider personal query logs instead of the aggregated query logs, which could return ranking tailored particularly to the tastes of the user [122].

In this study, we use a two-months time window to construct the query contexts $\mathbb{Q}_{ext}$ and $\mathbb{Q}_{adj}$ from the query logs. However, a smaller time window may reflect more up-to-date trends of search intents for the same query. Furthermore, query extensions and adjacent queries can be weighted according to their temporal distances to the underspecified query. Another follow-up question is: given the importance of ordering, which type of the adjacent queries work better for *Q-Rank*, those that precede or follow the underspecified query?

Finally, we conclude our discussion with a comment about the implementation of Q-Rank, which currently uses only the document snippets (rather than the full-text of the retrieved documents). This choice makes it very efficient without adding much complexity to the ranking module of a search engine, paying a much smaller overhead compared to using the full-texts. However, it also introduces a dependency on the quality of the snippets generated by a search engine and may increase its vulnerability to web spam. Nevertheless, only the top N search results are considered as re-rank candidates.

4.6 Summary

We study the implication of the collective user feedback on search query reformulations for relevance ranking improvement. We describe a method *Q-Rank* to leverage such implicit feedback embedded from the search engine query logs to re-rank the initial search results, and evaluate the performance of our proposal

against human editorial judgments. We also perform a series of comprehensive experiments to empirically select the best parameters of the algorithm. Evaluation results show that our re-ranking algorithm improves the quality of relevance ranking for a large-scale commercial Web search engine for 82% of the queries.

Chapter 5

Further Evaluation on Q-Rank

In this chapter, we further report on the evaluation of our proposal in Chapter 4, *Q-Rank*, against the query-log based authority analysis algorithm proposed by Luxenburger et al. [84]. To avoid potential naming confusion, we denote Luxenburger's algorithm by *QL* for the rest of this thesis.

We first introduce details about the *QL* algorithm proposed by Luxenburger et al. as our baseline. We then describe our data collection and preparation process for both the baseline and the *Q-Rank* algorithm, as well as the experimentation methodology and the evaluation metric. We present and discuss the comparison results towards the end of this Chapter, complete with two case studies.

5.1 Query-log based authority analysis

Luxenburger et al. proposed *QL* as an extension to the original PageRank algorithm, by incorporating the information of query reformulations, user clicks, and the similarity between queries and documents [84]. By extending the original *random surfer* graph, each vertex $\mathbf{v} \in \mathcal{V}$ now represents either a query $\mathbf{q} \in \mathcal{Q}$ or a webpage $\mathbf{p} \in \mathcal{P}$. Furthermore, there are five types of links among these vertices:

- $link(\mathbf{u}, \mathbf{v})$, where $\mathbf{u} \in \mathcal{P}, \mathbf{v} \in \mathcal{P}$. This denotes an hyperlink pointing from a page represented by vertex $\mathbf{u}$ to another page $\mathbf{v}$.
- $ref(\mathbf{u}, \mathbf{v})$, where $\mathbf{u} \in \mathcal{Q}, \mathbf{v} \in \mathcal{Q}$. This indicates that a user reformulates the query represented by vertex $\mathbf{u}$ to another query $\mathbf{v}$.

- $clk(\mathbf{u}, \mathbf{v})$, where $\mathbf{u} \in \mathcal{Q}, \mathbf{v} \in \mathcal{P}$. This indicates that a user issues a query represented by $\mathbf{u}$ and clicks on a page $\mathbf{v}$, which is among the search results returned.

- $qsim(\mathbf{u}, \mathbf{v})$, where $\mathbf{u} \in \mathcal{Q}, \mathbf{v} \in \mathcal{Q}$. This denotes the similarity between a pair of queries represented by $\mathbf{u}$ and $\mathbf{v}$.

- $psim(\mathbf{u}, \mathbf{v})$, where $\mathbf{u} \in \mathcal{P}, \mathbf{v} \in \mathcal{P}$. Like the definition of $qsim(\mathbf{u}, \mathbf{v})$, this denotes the similarity between a pair of web pages represented by $\mathbf{u}$ and $\mathbf{v}$.

Thus, the first three types of links are uni-directional (i.e., arcs) whereas the last two are bi-directional (i.e., edges). Figure 5.1, adapted from Luxenburger et al. [84], shows an example of such a random walk graph. There are three query vertices and four page vertices. *Query A* triggers a query-result click on *URL D*, and is also reformulated to *Query B*. *Query B* triggers a user click on *URL A*. On the other hand, *Query A'*, a search query similar to *Query A*, triggers a user click on *URL C*. *URL A* has an outgoing hyperlink to *URL B*. *URL B* and *URL C* are considered similar web pages, and both have incoming hyperlinks from *URL D*. Additionally, *URL B* also links back to *URL D*.

Intuitively, a random walker starts on either a query vertex $\mathbf{q}$ or a page vertex $\mathbf{p}$, and will perform the random walk in the following fashion. From any vertex, the walker can either jump to another vertex uniformly randomly, or follow an outgoing link in the following fashion. When the random walker is on a query vertex (e.g., *Query A*) and follows an outgoing link, it can visit a page vertex (e.g., *URL D*) via a query-result click arc $clk()$, another query vertex (e.g., *Query A'*) via a query similarity edge $qsim()$, or another query vertex (e.g., *Query B*) via a query reformulation arc $ref()$. On the other hand, when the random walker is on a page vertex (e.g., *URL B*) and follows an outgoing link, it can visit another page vertex (e.g., *URL D*) via a hyperlink arc $link()$, or another page vertex (e.g., *URL C*) via a similarity edge $psim()$. We refer the readers to Chapter 4 in the original paper [84] for the formal definition of the *QL* algorithm.

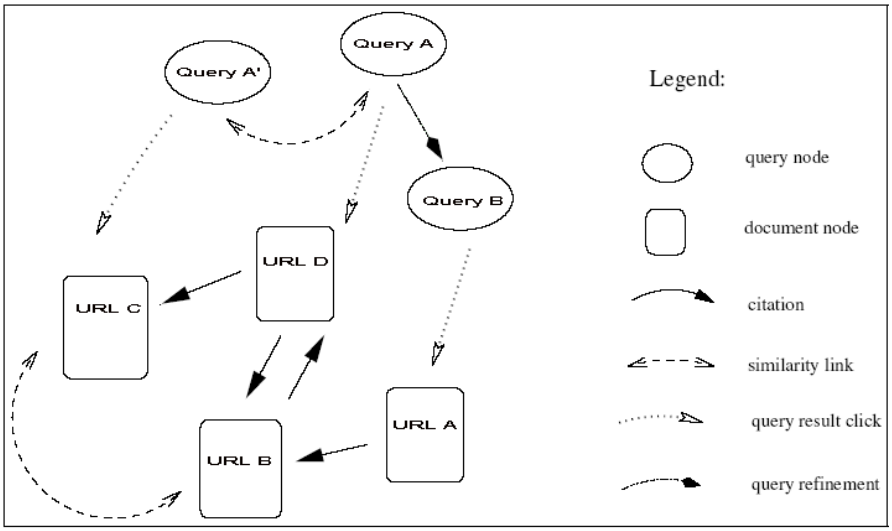

Figure 5.1. An example of the random walk graph formulation. Shown here are three different query vertices and four different page vertices. Additionally, there are three different types of arcs representing outgoing hyperlinks ($link()$), query refinement relationship ($ref()$), and query result clicks ($clk()$). And there are two different types of edges representing query similarity ($qsim()$) and page similarity ($psim()$).

5.2 Data Collection and Preparation

We collected data and constructed the random walk graph in the following fashion. First, we randomly sampled approximately 18,000 Web pages and the associated 1,000 unique search queries from the Yahoo! Search query logs of 15 days, from 1/1/2008 to 1/15/2008. These Web pages were among the search results of the search queries. We denote the set of these Web pages $\mathcal{P}_t$ and the set of the queries $\mathcal{Q}_t$. Next, we sampled approximately 150,000 query reformulations in the same search sessions of the aforementioned 1,000 queries, and about 110,000 unique Web pages that received at least one user click for the queries from the logs. Here, search sessions were separated by 30 minutes or more of idle time. And query reformulations were defined as the immediately subsequent queries issued by the same user in the same search session. We denote the set of query reformulations $\mathcal{Q}_r$ and the associated set Web pages $\mathcal{P}_r$. Finally, let the set of all Web pages $\mathcal{P} = \mathcal{P}_t \bigcup \mathcal{P}_r$, the set of all search queries $\mathcal{Q} = \mathcal{Q}_t \bigcup \mathcal{Q}_r$, and the set of vertices in the random walk graph $\mathcal{V} = \mathcal{P} \bigcup \mathcal{Q}$. Note that eventually, we were interested in scoring and ranking those Web pages in $\mathcal{P}_t$ for the corresponding queries in $\mathcal{Q}_t$.

Table 5.1 shows three examples of search sessions. In the first search session, the original query issued by the user, *domino pizz*, contained a misspelled query term, and user didn't click on any of the search results. Subsequently, a reformulated query *domino's pizza* triggered a user click on the desired webpage *www.dominos.com*. Similarly, in the second session, the original query *napa county sales tax* was later reformulated into *napa county retail sales tax*, and the user clicked on one search result *www.napacountygeneralplan.com/.../nci_report.pdf*. In the third session, the query *sample editing paragraphs lesson plans* seemed to have returned a number of desirable results, and the user clicked on three of them.

query	**webpage**	**action**
domino pizz	N/A	N/A
domino's pizza	*www.dominos.com*	click
napa county sales tax	N/A	N/A
napa county retail sales tax	*www.napacountygeneralplan.com/.../nci_report.pdf*	click
sample editing paragraphs lesson plans	*www.readwritethink.org/...view.asp?id=122*	click
sample editing paragraphs lesson plans	*www.readwritethink.org/...&engagement=0*	click
sample editing paragraphs lesson plans	*www.geocities.com/Athens/4868/teach.html*	click

Table 5.1. Examples of three search sessions. The first and the second search sessions contained query reformulations which led to user clicking on one of the search results. In the third session, the query *sample editing paragraphs lesson plans* triggered user clicks on multiple search results.

5.2.1 Random Walk Graph Construction for *QL*

To compute the similarity between a pair of queries $qsim(\mathbf{q}, \mathbf{q}')$ or a pair of webpages $psim(\mathbf{p}, \mathbf{p}')$, the following approach was used. First, for each webpage in our data collection, we employed a Maximum Entropy classifier [134] to classify it into the 150 high-level topics defined in Yahoo! Directory[1]. Yahoo! Directory is

[1]http://dir.yahoo.com/

a hierarchical taxonomy that covers a broad spectrum of topics, such as animal, news, photography, and sports. Table 5.2 shows part of the topical taxonomy used for webpage classification.

arts/humanities/literature
business_and_economy/finance_and_investment
computers_and_internet/programming_and_development
entertainment/movies_and_film
government/military
health/diseases_and_conditions
news_and_media/weather
recreation/automotive
science/astronomy
social_science/economics
society_and_culture/families

Table 5.2. Examples of the Yahoo! Directory topical taxonomy used for webpage classification.

Specifically, we associated each webpage with a topical vector of 150 components, each component a real number $\kappa \in [0, 1]$ denoting the likelihood of this webpage belonging to a corresponding topic in the Yahoo! Directory, generated by the MaxEnt classifier, in the form of

$$\mathbf{p} = \langle \kappa_1, \kappa_2, \cdots, \kappa_{150} \rangle.$$

Then, for a pair of webpages $\mathbf{p}, \mathbf{p}'$, we calculated their similarity as the cosine similarity of their topical vectors

$$psim(\mathbf{p}, \mathbf{p}') = \cos(\mathbf{p}, \mathbf{p}') = \frac{\mathbf{p} \cdot \mathbf{p}'}{\|\mathbf{p}\| \|\mathbf{p}'\|}.$$

Likewise, for each query in our dataset $\mathbf{q}$, we retrieved from Yahoo! Search the top 100 search result pages $\{\mathbf{p}_q\}$, and aggregated the topical vectors of these search result pages to derive the topical vector of the query. We then calculated the similarity between a pair of queries as the cosine similarity of the corresponding topical vectors.

$$\mathbf{q} = \frac{\sum \mathbf{p}_q}{150},$$

and

$$psim(\mathbf{q}, \mathbf{q}') = \cos(\mathbf{q}, \mathbf{q}').$$

In order to achieve a high confidence in our similarity measure, we applied a threshold τ for the pair-wise similarity. That is, a pair of queries or pages shared a similarity edge (i.e., $qsim()$ or $psim()$) only if the cosine similarity s of their topical vectors exceeded τ. And for all such pairs, their similarity value was further normalized into $[0, 1]$ for use in the ranking algorithm:

$$psim = \frac{s - \tau}{1 - \tau}, qsim = \frac{s - \tau}{1 - \tau}.$$

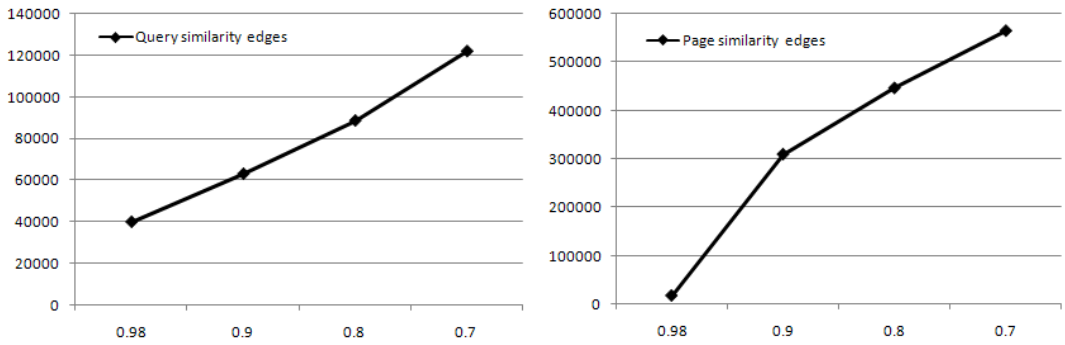

Figure 5.2. The number of query-query and page-page similarity edges increased dramatically as the similarity threshold τ dropped.

The decision to apply a higher similarity threshold was based on the following reasons. First, without any thresholding, the complexity of the pair-wise similarity computation for queries and pages was $O(n^2)$, and on our random walk graph it translated into hundreds of millions of query-query similarity edges and billions of page-page similarity edges. This would make the computation extremely expensive. Figure 5.2 shows the number of similarity edges for the similarity threshold values τ at 0.98, 0.9, 0.8, and 0.7. The number of similarity edges grew by close to 40% on average for every 0.1 drop in the similarity threshold value. Second, because the topical taxonomy applied to classify the pages (Table 5.2) was of high level (e.g., *literature*, *military*), a high similarity threshold will ensure the confidence in claiming the pair was somewhat topically similar (thus creating an edge in between).

For example, for a pair of irrelevant queries, *coastal living* and *carpenter gmac*, their similarity was 0.95 when $\tau = 0.6$, 0.8 when $\tau = 0.9$, and 0.01 when $\tau = 0.98$.

Clearly, a higher similarity threshold made more sense in this case. We manually inspected random samples of query-query and page-page pairs at different similarity thresholds and the observations were consistent.

We also empirically evaluated different similarity threshold values for QL. When $\tau = 0.7$, NDCG was 0.645 and DCG@5 was 5.835; when $\tau = 0.98$, NDCG was 0.646 and DCG@5 was 5.838. Given the (slight) gains at a higher similarity threshold and the benefit of dramatically less expensive computation, we decided to choose $\tau = 0.98$ as the similarity threshold.

See Table 5.3 for examples of the query pairs and their topical similarity. In total, we created 17,108 implicit edges (i.e., $psim()$) for pairs of similar webpages, and 39,996 implicit edges (i.e., $qsim()$) for pairs of similar search queries. Note that the smaller number of edges relative to the total number of queries and pages in the dataset was due to the similarity thresholding.

query A	**query B**	*qsim*
digital photography basics	*digital photography tips*	0.999
childhood leukemia	*leukemia*	0.982
hp ipaq 6500	*hp ipaq rw6815*	0.971
linkin park song	*linkin park lyrics*	0.889
marriott airport	*marriott dulles*	0.646
diana movies	*office imdb*	0.507
safari animal clinic	*pine snakes*	0.422
stars tv	*friends episodes*	0.347
travel friends	*marriott brooklyn*	0.273
seat cushion covers	*2000 buick century custom*	0.155
lacrosse antenna	*discount bose*	0.003
coastal living	*carpenter gmac*	0.001

Table 5.3. Examples of queries and their pair-wise similarity.

Next, we extracted all the clicked webpages for both the original and the reformulated queries from the search engine logs. For each pair of the corresponding query vertex $\mathbf{q}$ and the webpage vertex $\mathbf{p}$, we created an unweighted arc $clk(\mathbf{q}, \mathbf{p})$. Table 5.4 depicts sample pairs of queries and the click webpages in our data collection.

query	clicked webpage
capital one	capitalone.com
capital one	auto-credit.com
2008 election poll	en.wikipedia.org/wiki/u.s._presidential_election,_2008
2008 election poll	www.cnn.com/election/2008
2008 election poll	news.yahoo.com/election/2008
marriott	en.wikipedia.org/wiki/marriott_corporation
marriott	spas.about.com/od/nevada/fr/jwmarriottvegas.htm

Table 5.4. Examples of queries and their clicked webpages extracted from the search engine logs.

To capture the reformulation relationship among queries in the search sessions, we created an arc $ref(\mathbf{q}, \mathbf{q}')$ from a query vertex $\mathbf{q}$ to another one $\mathbf{q}'$, for each pair of reformulations we extracted from the query logs. Here, search sessions were arbitrarily separated by 30 minutes of inactivity. We then computed the reformulation frequency ratio $r_f(\mathbf{q}, \mathbf{q}')$ as the arc weight. Formally, let $freq(\mathbf{q}, \mathbf{q}')$ denote the number of times such a reformulation is observed in the logs, and $\mathbf{V}_r(\mathbf{q})$ denote the set of reformulated queries of $\mathbf{q}$, we have

$$r_f(\mathbf{q}, \mathbf{q}') = \frac{freq(\mathbf{q}, \mathbf{q}')}{\sum_{\mathbf{q}' \in \mathbf{V}_r(\mathbf{q})} freq(\mathbf{q}, \mathbf{q}')}.$$

See Table 5.5 for examples of the query reformulations for the queries *2008 election poll* and *marriott*, and the corresponding arc weights.

query	reformulation	r_f
2008 election poll	*2008 election poll results*	0.156
2008 election poll	*us 2008 election polls*	0.139
2008 election poll	*2008 election poll republicans*	0.093
2008 election poll	*2008 election poll vote*	0.051
marriott	*marriott hotels*	0.458
marriott	*marriott rewards*	0.091
marriott	*jw marriott*	0.018
marriott	*marriott jobs*	0.005

Table 5.5. Examples of query reformulation pairs and the corresponding arc weights. Note that the list of reformulations is not exhaustive, thus the arc weights here do not sum up to one.

Finally, for each pair of webpages represented by vertices $\mathbf{p}$ and $\mathbf{p}'$, we created

an unweighted arc $link(\mathbf{p}, \mathbf{p}')$ if and only if there exists a hyperlink pointing in the same direction.

5.2.2 Characteristics of the Random Walk Graph

In this section, we describe the characteristics of the final random walk graph constructed for computing QL.

Property	Value
# Vertices	272,272
# Arcs	272,849
# Edges	57,104
Giant component percentage	99.78%
Diameter	31
Average path length	7.485

Table 5.6. Characteristics of the final random walk graph. The giant component contained 271,673 vertices, reaching about 99.78% of the whole graph.

Table 5.6 shows the basic properties of the graph. Most notably, the connectivity of the graph was measured by the *reachability* of the giant component, which contained more than 99% of all the vertices. On the other hand, the average path length, i.e., the average length of the shortest path between a pair of two vertices, is 7.485. This coincided with the intuition of the "six degrees of separation". The diameter of the graph, i.e., the length of the longest shortest path between two pairs of vertices, is 31.

5.2.3 Parametrization for *QL*

The QL algorithm had two tunable parameters. The first parameter α specified the probability, when the random surfer was to follow an outgoing link, to follow an *explicit* link ($link()$, $ref()$, or $clk()$) instead of an *implicit* link ($qsim()$ or $psim()$). The second parameter β specified the probability, when the random surfer was to perform a random jump, to jump to a query vertex $\mathbf{q}$ instead of a page vertex $\mathbf{p}$.

We evaluated five variations of QL parametrization on the random walk graph (Table 5.7). Figure 5.3 (a) and (b) show the corresponding DCG@1 and DCG@5 for these settings. There were three observations. First, in settings where we

Notation	α	β
A1/B0	1.0	0.0
A1/B0.5	1.0	0.5
A0/B1	0.0	1.0
A0.5/B0.5	0.5	0.5
A0.5/B1	0.5	1.0

Table 5.7. Definition of the five parameter settings evaluated for *QL*.

incorporated the two implicit similarity relationship ($qsim()$ and $psim()$) into the random walk process (i.e, A0.5/B0.5 and A0.5/B1), the algorithm performed better than the settings solely based on explicit links (i.e., A1/B0 and A1/B0.5). The only exception was the setting A0/B1, which performed the worst. An intuitive explanation was that when $\alpha = 0.0$, the random surfer would only follow the implicit similarity links and completely ignore the explicit links, and would suffer from the lack of user feedback that was captured by the clicks and the query reformulations. Second, the performance of *QL* was less sensitive to the parameter β, which was highly dependent on the ratio of the query vertices to the page vertices in the graph. Third, given the same α, the DCG performance decreased slightly in both metrics as β increased, which indicated that it was desirable to have a lower likelihood of a query vertex being the destination of a uniformly random jump.

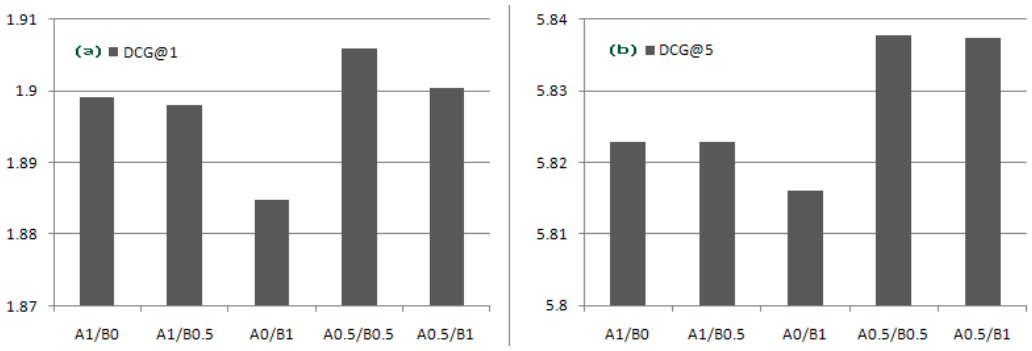

Figure 5.3. Comparison of (a) DCG@1 and (b) DCG@5 for different parametrization of the *QL* algorithm. First, by incorporating the implicit similarity links ($qsim()$ and $psim()$) into the random walk process (i.e, $\alpha \neq 1.0$), the algorithm performed better than solely based on explicit links (i.e., $\alpha = 1.0$). Second, the performance of *QL* was less sensitive to the parameter β, which was highly dependent on the number of the query vertices. These two observations were consistent with the empirical findings in Luxenburger et al. [84].

These observations were consistent with the empirical findings in Luxenburger et al. [84].

Based on the aforementioned evaluation, in our implementation of *QL* we chose the best-performed parameter setting A0.5/B0.5 (i.e., $\alpha = 0.5, \beta = 0.5$). In fact, this was the same as the *"QRank-Q/D-Sim"* setting described in the paper by Luxenburger et al. [84], which was also reported to have performed either the best or the second best across all the evaluations.

5.2.4 Remarks on expanded neighborhood graphs

In this section we studied the impact of the hyperlinked neighborhood on the performance of *QL*. Let us denote the aforementioned random walk graph by $\mathcal{G}$. We evaluated *QL* with the parameter setting A0.5/B0.5 on two *expanded* random walk graphs, based on the hyperlinked neighborhood of the result set in $\mathcal{G}$.

First, we followed the outlinks of the 18,000 pages in the initial result set $\mathbb{D}_{\mathbf{q^t}}$, and crawled 514,355 unique pages. These pages belonged to what we called the *1-hop-away neighborhood* of the result set. We expanded the random walk graph $\mathcal{G}$ by including the pages in this *1-hop-away neighborhood* and the hyperlinks, and denote the expanded graph by $\mathcal{G}_{1h}$.

Second, we followed further the outlinks of the pages in the *1-hop-away neighborhood* and crawled 10,383,159 unique pages in what we called the *2-hop-away neighborhood* of the result set. Note that the *2-hop-away neighborhood* was a superset of the *1-hop-away neighborhood*, as it contained both the 1-hop-away pages and the 2-hop-away pages. To reduce the size of the graph, we pruned the *2-hop-away neighborhood* by removing pages without outlinks back into the result set and their corresponding ancestors in the *1-hop-away neighborhood*. In the end, the pruned *2-hop-away neighborhood* contained 1,238,286 pages, roughly one-tenth of its original size. We then expanded $\mathcal{G}$ by including this pruned *2-hop-away neighborhood* and denote the expanded graph by $\mathcal{G}_{2hp}$.

We applied *QL* with the parameter setting A0.5/B0.5 on both $\mathcal{G}_{1h}$ and $\mathcal{G}_{2hp}$, and compared its performance with that when applied on $\mathcal{G}$ (Table 5.8).

Not surprisingly, expanding the graph to the *1-hop-away neighborhood* didn't offer much benefit, which could be explained by the fact that the majority of the

zero-out-degree vertices in the neighborhood were sink vertices for the random surfer and did not contribute much back to the pages in the result set. This observation was consistent with the comments made by Luxenburger et al. [84] and Brin et al. [17].

On the other hand, expanding the graph to the pruned *2-hop-away neighborhood* offered some benefit by considering the connectivity of the larger neighborhood. However, the improvements were only 1.9% on DCG@1 (significant) and 0.9% on DCG@5 (not significant), and couldn't justify the much more additional computation costs caused by the larger graph size.

Graph	**DCG@1**	**DCG@5**
$\mathcal{G}_{1h}$	1.9 (-0.26%, p=.817)	5.83 (-0.05%, p=.735)
$\mathcal{G}_{2hp}$	1.94 (+1.81%, p=.089)	5.89 (+0.92%, p=.013)

Table 5.8. The comparison of *QL* performance on the *1-hop-away neighborhood* graph $\mathcal{G}_{1h}$ and the pruned *2-hop-away neighborhood* $\mathcal{G}_{2hp}$, using the original random walk graph $\mathcal{G}$ as the baseline. Not surprisingly, expanding the graph to the *1-hop-away neighborhood* didn't offer much benefit. Even though the pruned *2-hop-away neighborhood* did offer some benefit, the DCG improvements were not significant enough to justify the much more additional computation costs.

5.2.5 Data Preparation and Parameter Choice for *Q-Rank*

In this section, we describe how we prepare the query reformulation data for use in the *Q-Rank* algorithm.

To conform to the *Q-Rank* annotations used in Chapter 4, we denote each of the 1,000 queries as the *initial query* $\mathbf{q^t}$, and use $\mathbb{Q}_t$ to denote the set of reformulations randomly sampled from the logs during a time window of $t = 15$ days. Thus, $\mathbf{q} \in \mathbb{Q}_t$ denotes a single reformulation instance for the query $\mathbf{q^t}$. Furthermore, we denote the set of search results for $\mathbf{q^t}$ as $\mathbb{D}_{\mathbf{q^t}}$.

Given an initial query $\mathbf{q^t}$, we categorized $\mathbf{q} \in \mathbb{Q}_t$ into two types, *query extensions* ($\mathbb{Q}_{adj}$) and *adjacent queries* ($\mathbb{Q}_{ext}$), following the same definitions described in the *Q-Rank* algorithm in Chapter 4. Table 5.9 shows the top reformulations of the query *charter club*, their types, and the frequencies. We were then able to compute the reformulation frequency $qf(\mathbf{q}), \forall \mathbf{q} \in \mathbb{Q}_{adj} \bigcup \mathbb{Q}_{ext}$, as well as the normalized reformulation frequency in Equation 4.3. We also computed the $tf \cdot idf$

scores [108] for each pair of the reformulated query $\mathbf{q}$ and the document $\mathbf{d} \in \mathbb{D}_{\mathbf{q^t}}$.

reformulation	type	frequency
charter club bedding	extension	162
charter club jewelry	adjacency	42
charter club slippers	extension	39
charter club slippers	adjacency	20
charter club pajamas	adjacency	14
macy's charter club	extension	10
charter club pillows	extension	8
marrakech charter club	adjacency	3

Table 5.9. Examples of the most frequent reformulations of the query *charter club*, their types, and the number of occurrences in the query logs.

Furthermore, we set the size of the query context sets $|\mathbb{Q}_{ext}|$ and $|\mathbb{Q}_{adj}|$ to be 20, which was consistent with the evaluations discussed in the previous Chapter.

Since we randomly sampled the search result pages instead of drawing from the top results for a given query, for any *initial* query $\mathbf{q^t}$, we set the number of unchanged top results $u = 0$, the ranking range $n = |\mathbb{D}_{\mathbf{q^t}}|$, the number of re-rank candidates $c = |\mathbb{D}_{\mathbf{q^t}}|$, and the $bias = false$, i.e., $R(\mathbf{d}) = 1, \forall \mathbf{d} \in \mathbb{D}_{\mathbf{q^t}}$.

Additionally, we set $\gamma = 0.5$, which had shown the largest relevance improvement in our previous experiments described in Chapter 4.

5.3 Evaluation

5.3.1 Evaluation Methodology and Metric

We evaluated *Q-Rank* by comparing its effectiveness with that of *QL* as the baseline algorithm.

To quantify our relevant measurement, a team of editors were employed to rate the relevance between the search queries and the webpages on a five-point scale: *perfect*, *excellent*, *good*, *fair*, *bad*. These editors were professionally trained to evaluate the search results' relevance for the given queries. Their relevance judgments were subsequently projected into integer ratings from 4 to 0, and used to compute the Discounted Cumulative Gain (DCG) metric [58, 60] which was also used in the other evaluations documented in this thesis.

For each of the 1,000 *initial* queries $\mathbf{q^t}$, we computed using *Q-Rank* and *QL* two ranking scores for each of the search result pages in $\mathbb{D}_{\mathbf{q^t}}$, and generated two corresponding ranked lists in descending order of the scores. We then calculated the NDCG metric, the DCG metric for the top-ranked document (DCG@1), the top-five ranked documents (DCG@5), and the top-ten ranked documents (DCG@10) for both lists. Using the NDCG and the DCG scores of the ranked list produced by *QL* as the baseline, we were then able to quantify the relevance improvement of *Q-Rank* over *QL*.

	NDCG		**DCG@1**		**DCG@5**		**DCG@10**	
Queries	*QL*	*Q-Rank*	*QL*	*Q-Rank*	*QL*	*Q-Rank*	*QL*	*Q-Rank*
Overall	0.68	+3.5% (p=.001)	2.52	+29.9% (p=.043)	7.44	+17.5% (p=.01)	11.6	+11.3% (p=.007)
One word	0.66	+5.3% (p=.024)	2.43	+45.8% (p=.218)	6.42	+34.9% (p=.07)	10.1	+21.4% (p=.06)
Two words	0.71	+2.6% (p=.134)	3.17	+23.4% (p=.241)	8.73	+9.2% (p=.183)	13.4	+6.6% (p=.163)
Three+ words	0.7	+2.9% (p=.161)	2.13	+22.4% (p=.296)	7.55	+13.5% (p=.334)	11.8	+5.8% (p=.473)

Table 5.10. Relevance improvements of *Q-Rank* over Luxenburger's query-log based authority algorithm. Overall improvements were statistically significant for all three metrics, and one-word queries had more gains compared with longer queries. Improvements on DCG@5 and DCG@10 for one-word queries were nearly significant.

5.3.2 Results and Discussion

In this section we report and discuss the evaluation results comparing the relevance ranking produced by *Q-Rank* and by *QL*.

Table 5.10 depicts the relative DCG improvement of *Q-Rank* over *QL* with the corresponding t-test p-values. Here, let DCG_{qr} denote the DCG of the ranking produced by *Q-Rank*, and DCG_{ql} denote the DCG of the ranking produced by *QL*, then the hypothesis was $DCG_{qr} \geq DCG_{ql}$. In addition to showing the overall improvement for all queries, the results were further organized by the various query lengths.

Q-Rank proved more beneficial for shorter queries (esp. one-word queries) compared with longer queries, which was consistent with our intuition that the

longer queries had fewer reformulated extensions available in the query logs.

The complexity of *Q-Rank* is $O(n \log n)$, compared with $O(n^2)$ of *QL*. However, in practise *Q-Rank* requires runtime computation of the relevance scores between the pages and the query context, while *QL* can typically compute the scores offline.

5.3.3 Case Studies

Case Study: *interceramic*

Table 5.11 (a) shows the search result rankings generated by *Q-Rank* and *QL* for the query *interceramic*. One most frequent query extension was *interceramic usa* which accounted for more than 11% of all the reformulations in the search logs. With a strong emphasis on capturing the frequent query reformulations for ranking the webpages, *Q-Rank* was able to rank at the top the two most relevant pages, *www.interceramicusa.com/us/main.asp* and *www.interceramicusa.com*, while *QL* ranked them at the 16th and the 17th.

Case Study: *james wood chevrolet*

Another example of improved relevance ranking was for the query *james wood chevrolet* (Table 5.11 (b)). In this case, *Q-Rank* introduced more diversified results into the top of the ranking. As we inspected the query and the webpages returned, it turned out that there were at least two James Wood Chevrolet auto dealers: one located in Denton, TX and the other in Decatur, TX. Actually, the query extensions *james wood chevrolet denton* and *james wood chevrolet decatur* accounted for 76% and 20% of the reformulations, respectively. *Q-Rank* was able to leverage on the frequent extensions so that the official homepages of the auto dealership in Decatur (*nitra.gmpsdealer.com/james-w-commercial/en_US*[2] and *www.jameswooddecatur.com*) was ranked as the 3rd and the 10th, respectively.

[2]It redirected to *www.commercialtrucksdecatur.com*.

(a)	**Query:** ***interceramic***			
	Q-Rank		***QL***	
rank	webpage	rating	webpage	rating
1	www.interceramicusa.com/us/main.asp	excellent	www.arcat.com/specwizard/09300int/?coid=33309	fair
2	www.interceramicusa.com	perfect	www.barefootfloor.com/ceramic-tiles/interceramic/...	good
3	www.designbiz.com/net4/CompanyProductCollection...	fair	www.barefootfloor.com/woodlands-interceramic-...	fair
4	www.designbiz.com/biz/BrandProfile.asp?...	good	www.bid4floors.com/interceramic-...	fair
5	www.designbiz.com/biz/BrandProfile.asp?...	good	www.bid4floors.com	fair
6	www.bid4floors.com	fair	www.answers.com/topic/internacional-de-ceramica-...	good
7	www.arcat.com/specwizard/09300int/?coid=33309	fair	www.designbiz.com/biz/BrandProfile.asp?...	good
8	www.barefootfloor.com/woodlands-interceramic-...	fair	www.designbiz.com/biz/BrandProfile.asp?...	good
9	www.aecinfo.com/1/company/04/80/91/...1	good	www.designbiz.com/net4/CompanyProductCollection...	fair
10	www.nextag.com/interceramic/search-html	good	www.hoovers.com/interceramic,-inc./...	good
(b)	**Query:** ***james wood chevrolet***			
	Q-Rank		***QL***	
rank	webpage	rating	webpage	rating
1	www.jameswooddenton.com/ContactUsForm	good	www.jameswooddenton.com/PreOwnedVehicleSearch	good
2	www.jameswooddenton.com/PreOwnedVehicleSearch	good	www.jameswooddenton.com/MiscPage_3	good
3	nitra.gmpsdealer.com/james-w-commercial/en_US	excellent	www.jameswooddenton.com/ContactUsForm	good
4	www.jameswooddenton.com/VehicleDetails/1022007804	perfect	www.autoweb.com/cardealers.cfm/texas/Krum/Chevrolet	good
5	www.jameswooddenton.com/MiscPage_3	good	nitra.gmpsdealer.com/james-w-commercial/en_US	excellent
6	www.autoweb.com/cardealers.cfm/texas/Krum/Chevrolet	good	www.pegasusnews.com/places/james-wood-chevrolet	fair
7	sports.yahoo.com/nascar/players	bad	www.jameswooddenton.com/VehicleDetails/1022007804	perfect
8	http://www.jameswood.com/ContactUsForm	good	sports.yahoo.com/nascar/players	bad
9	www.pegasusnews.com/places/james-wood-chevrolet	fair	www.autotrader.com/dealers/dda/inventory.jsp?...	good
10	www.jameswooddecatur.com	excellent	www.jameswood.com	excellent

Table 5.11. (a) Side-by-side comparison of the two search result rankings produced by *Q-Rank* and *QL* for the query *interceramic*. The duplicate entries from *www.designbiz.com* was due to the same page being sampled twice. (b) Side-by-side comparison of *Q-Rank*'s and *QL*'s search result rankings for the longer query *james wood chevrolet*.

Chapter 6

Mining User Clicks for Geographic Interests Inference

In the first half (Chapters 3 and 4) of the dissertation, we have studied a model of user feedback in collaborative ranking, and described methods to leverage such feedback to improve relevance ranking. In the second half (Chapters 6 and 7), we will investigate the usage of feedback in two verticals: geographic information retrieval systems and digital libraries.

We have witnessed the recent surge in the volume of search queries that explicitly or implicitly express users' geographical interests. More than one tenth of the search queries contain a *placename* [51, 112]. And about 10% of the query reformulations containing a placename involve a *geocorrection* (i.e., the user either spell-corrects or explicitly disambiguates the placename) [66].

To accurately infer users' locality preference becomes an increasingly important yet challenging problem. Santos and Chaves examined the semantics of place names in geographic queries [113]. According to their findings, location names are often ambiguous so that good entity recognition is important, and usually dependent on the context of usage so that inference of the implicit knowledge is required. While some queries (e.g., *"auto repair stanford"*) contain an explicitly location qualifier (*"stanford"*), other queries may come with spatial ambiguities. For example, a query *"bay bridge"* may refer to the San Francisco-Oakland Bay Bridge in California, the Chesapeake Bay Bridge in Maryland, or the Escambia Bay Bridge in Florida. Similarly, a query *"chicago pizza"* may refer to *chicago*

style pizza or *pizza stores in Chicago*, and for the same reason, the fact that a Web page contains both the terms *"chicago"* and *"pizza* doesn't help resolve the ambiguity, either.

In the above and many other similar scenarios, it is clear that syntactic information alone is not a good indicator to infer users' geographical interests. This observation motivates us to exploit the implicit user feedback such as their clicks to model the locality preferences of the users.

In contrast to a large body of existing work on modelling users' geographical interests based on syntactic information, we study two click-based models for inferring the geographical interests of the search engine users. We exploit the geographical distributions of user clicks, and empirically demonstrate in this paper that such distributions shed light on the geographical variations in users' interests for any given query or Web page. Specifically, we address the following three problems:

- Given a large number of search queries or Web pages, how do we model their geographical specificity?
- For queries and Web pages that are identified as having a *local flavor*, how do we detect and disambiguate the underlying geographical location(s) of interests?
- How do we improve the search relevance ranking given the model of the distribution of *geographical interests*?

Organization The rest of this section is organized as follows. We introduce the dataset in Section 7.2, and visualize the correlations between the geographical distribution of user clicks and their locality preference in Section 6.2, and present two click-based models that capture such correlation in Section 6.3. We discuss two applications of these models: identifying geo-sensitive queries (Section 6.4) and improving search relevance ranking (Section 6.5). In Section 6.6 we present empirical evaluations. In Section 6.7 we discuss how to cluster and disambiguate search results based on the user click feedback. Finally, we summarize this chapter in Section 6.8.

Table 6.1. Definitions of Query Geo-Sensitivity

Sensitivity Category	Definition	Examples
Explicit	Queries about local business, organizations, product, and information of a particular location, which explicitly specify a geographical location.	*georgia department of revenue, nissan dealer in bay area, Seattle tax rate*
Implicit	Queries that implicitly specify a location with a popular or famous local business or landmark.	*dulles airport, broward community college, aspen grove shopping center, dumbarton bridge*
Local	Queries that are not specific to a particular location but local information is implicitly preferred; typically contain the name or type of a business, service, or organization without a specific geographical location.	*dentist, toyota dealer, AAA branch*
Non-Sensitive	Generic or navigational queries, and queries about information interesting to users regardless of their physical locations.	*yahoo, google, disneyland, new york times*

6.1 Dataset

We were granted access to the anonymized logs of a popular commercial search engine *Yahoo! Search*[1]. From the search query log, we sampled a set of 10,000 queries which were believed to be representative and covering a broad range of search topics. We then aggregated about 450M user clicks on these queries from the click logs from November 2007 to April 2008, which contained about 1.5M unique (query, URL) pairs. Each record consisted of the following fields: query, URL, rank of the URL in the search results, IP address, and the aggregated number of clicks which originated from this address. An example is (*"florida international university"*, *"http://www.stanford.edu/group/ree/reeusa03/posters/FIU.pdf"*, 58, 4ae96544, 1). Note that we collapse multiple clicks from the same search session into one.

After filtering the IP addresses of popular proxy servers and large ISP hubs, we

[1]http://search.yahoo.com

used a proprietary IP lookup database to map each IP address to a geographical location (i.e., latitude, longitude, city/town, county, state, zipcode, and country). In the previous example *"florida international university"*, the geographical location is (25.727, -80.235, Miami, Miami-Dade, FL, 33133, USA).

6.2 Visualize Geographic Interests Distribution

The *geo-sensitivity* of a query denotes that, to answer the query, Web pages that either have explicit / implicit association with certain geographical location(s) or are considered more relevant to users in certain geographical location(s) will be considered more *relevant*. To make it more concrete, we define four categories of geo-sensitivity for queries in Table 6. For the rest of this paper, we will follow these definitions. If a search query falls into either the *explicit* or the *implicit* sensitivity category, it is referred to as a *Geo-Sensitive Query* (*GSQ*), otherwise a *Non-Geo-Sensitive Query* (*NGSQ*).

Previously we discussed the disadvantage of relying solely on the syntactic information of a query to infer its geo-sensitivity. Assume that a user submits a query and the search engine returns a list of Web pages, he/she is more likely to click on Web pages that appear *more relevant*. For these *GSQ*s, whether a Web page is *geographically* relevant is an important factor of the overall relevance perceived by the user. Thus, given a search query, once we aggregate a significant number of user clicks and conduct a reverse-lookup of the IP addresses to find out where they come from, it is possible to learn about the locality preference of this query by *"following the herd"*. For example, a *GSQ "broward community college"* received the majority of user clicks originated from the Fort Lauderdale, FL area, which could indicate that this query is *more geographically sensitive* to the Fort Lauderdale area than anywhere else. Thus, we examine the following two hypothesis:

Hypothesis 1. *Given a search query or a Web page, the geographical distributions of user clicks it receives approximate the geographical distribution of users' interests on it.*

Hypothesis 2. *The geographical distributions of user clicks for* NGSQ*s and* GSQ*s*

have different patterns: the aggregated click distribution of NGSQs *resembles the user population distribution, while the distribution of* GSQs *differs from the population distribution.*

To examine the above hypothesis, we visualize the aggregated click records in our dataset using *click maps*. Here, the *click map* of a query visualizes the geographical locations of search users who have issued the query and also clicked on some of the results. Each red dot on the click map of a query represents the aggregated user clicks originated from the corresponding geographical location.

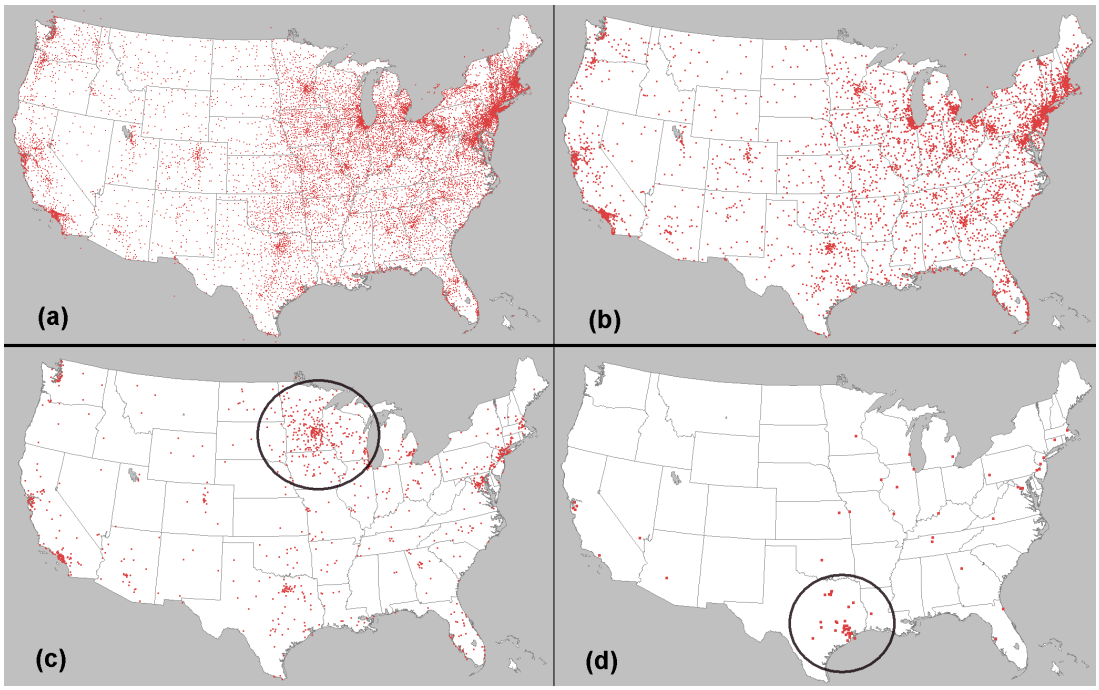

Figure 6.1. (a) The overall click map of randomly sampled 8.9K search queries. (b) The click map of query *"Google"*. (c) The click map of query *"Sun Country Airlines"*. (d) The click map of query *"93.7 Houston"*. Regions with unusually high density of clicks for the last two queries are indicated by the black circle.

Figure 6.1(a) shows a click map of 8.9K randomly drawn search queries, in which the geographical distribution of user clicks mostly follows the population distribution in the U.S. Compared with Figure 6.1(b) which shows a click map of the query *"Google"*, it is obvious that the two resemble each other, indicating that the query *"Google"* is not *geographically sensitive* to a particular location.

In contrast, Figure 6.1(c) and 6.1(d) show the click maps of two *geo-sensitive*

queries *"Sun Country Airlines"* and *"93.7 Houston"*, respectively. The spatial distributions of the user clicks for these two queries both display a strong deviation from the population distribution, and have unusually high density in some regions: *"Sun Country Airlines"* received more than 30% of its clicks coming from Minneapolis, MN (a hub of the airline), and *"93.7 Houston"* – a local radio-station in Houston – received more than 80% of its clicks from Houston, TX.

As we examined the click maps of more search queries, it was consistent that the geographical distribution of user clicks were usually quite different between *GSQs* and *NGSQs*.

6.3 Click-based Models of Geographic Interests

In this section, we study two models of the geographical distribution of users' interests based on historical clicks.

6.3.1 Vector model

We first define the set of geographical regions as $\mathbb{R} \doteq \{\mathbf{r}\}$, where $\mathbf{r}$ denotes a physical location of an appropriate granularity (i.e., how large the region is). The granularity can be either iteratively tuned or chosen arbitrarily.

Next, we define the set of queries as $\mathbb{Q} \doteq \{\mathbf{q}\}$, where $\mathbf{q} = \langle c_1, c_2, \cdots, c_k \rangle$ denotes a query. $\mathbf{q}$ is a k-dimension vector, and each dimension c_i represents *the probability of the query receiving user clicks from a geographical region* $\mathbf{r_i} \in \mathbb{R}$. Here c_i can be defined as

$$c_i = \frac{c_{r_i}}{\sum_{r_i \in \mathbb{R}} c_{r_i}} \cdot (1 + \log \frac{|\mathbb{Q}|}{|\{\forall \mathbf{q} : c_{r_i} \neq 0\}|}), \tag{6.1}$$

where c_{r_i} is the number of user clicks from r_i for $\mathbf{q}$.

The same model can also be applied to the set of Web pages as $\mathbb{D} \doteq \{\mathbf{d}\}$, where $\mathbf{d} = \langle c'_1, c'_2, \cdots, c'_k \rangle$ denotes a Web page, and each dimension c'_i denotes *the probability of the Web page receiving user clicks from a geographical region* $r_i \in \mathbb{R}$, with a definition of c'_i similar to c_i.

6.3.2 Maximum-likelihood model

Backstrom et al. proposed a generated model to derive the geographic center and spatial dispersion of search queries [6]. The model was based on the observation of the *view* event in the search logs: a user issues a query from an IP address which is mapped to a physical location.

In our work, we provide two extensions to this model. First, we extend this model to the *click* events recorded in the search logs: a user clicks on a certain Web page from an IP address mapped to a physical location. Our motivation is that compared with the action of issuing a query, user clicks are stronger feedback of endorsement from a relevance standpoint. Second, we adapt this model to both queries and Web pages, and are able to measure the proximity of the *"centers of interests"* of a given query and a given Web page as a relevance ranking feature.

Let $\mathbb{Q} \doteq \{\mathbf{q}\}$ denote the set of queries in the query logs. Each query $\mathbf{q}$ has a geographic center of "interests" $\mathcal{C}$, from where most of the user clicks for this query are observed, and the observed number of user clicks from anywhere else is in reverse proportion to its distance d away from $\mathcal{C}$. Assuming the distribution of user clicks decay exponentially further away from the center of "interests", then we model the probability p of observing a click by a random user at distance d from $\mathcal{C}$ as

$$p = C \cdot d^{-\alpha}, \tag{6.2}$$

where C is the observed number of user clicks originated from $\mathcal{C}$, and the exponent α is a spreading factor which specifies how fast the number of clicks decreases further away from $\mathcal{C}$.

Therefore, given a query $\mathbf{q}$ and the past user clicks for this query, we can infer the center of interests as well as the corresponding spreading factor in a maximum-likelihood fashion,

$$\arg\max p^m (1-p)^n, \tag{6.3}$$

where m is the number of observations of a click by the user at distance d from $\mathcal{C}$, and n is the number of observations otherwise. Algorithmically, the probability maximization process runs iteratively. The region from which the "center of interests" is to be found can be split into $M \times N$ grids, each of K degrees of latitude

and longitude. We start with coarse grids, and select the grid g_i that maximizes the probability according to Equation 6.3. Next, we further split g_i into $M \times N$ smaller grids of a finer granularity, and select the grid g_{ij} that maximizes the probability. This process continues until a desired granularity is reached, typically at the tenths of degrees of latitude and longitude.

Similarly, we can apply this model for each Web page $\mathbf{d} \in \mathbb{D}$ that has registered user clicks in the search logs.

6.4 Classify queries based on geo-sensitivity

Witnessing the distinctive patterns of geographical click distributions of *GSQs*, we discuss how to algorithmically derive the geo-sensitivity for queries under a classification scheme. Accepting Hypothesis 2, can we differentiate *GSQs* from *NGSQs*, and infer the locality preference of a query based on its geographical click distribution?

Without loss of generality, the task to identify the *geo-sensitivity* of a search query can be cast into a binary classification problem, formally defined as follows:

Definition *Given a set of queries* $\mathbb{Q} = \{\mathbf{q}\}$*, classify* $\mathbf{q}$ *into one of the two classes: (1) a class of* geo-sensitive queries *GSQ* ($GSQ \subseteq \mathbb{Q}$)*, and (2) a class of* non-geo-sensitive queries *NGSQ* ($NGSQ = \mathbb{Q} \setminus GSQ$)*.*

Vector Model. Let vector $\mathbf{p} = \langle d_1^p, d_2^p, \cdots, d_k^p \rangle$ denote the user population, with each dimension d_i^p represents the population in a geographical region r_i same as in $\mathbf{q}$. By accepting Hypothesis 2, we chose to approximate the population distribution with the aggregated click probability distribution of all the queries in $NGSQ$, i.e., the vector summation of all vectors $\mathbf{q}' \in NGSQ$ and normalized:

$$\mathbf{p} = \frac{\sum \mathbf{q}'}{|NGSQ|}, \forall \mathbf{q}' \in NGSQ \tag{6.4}$$

.

Given a query $\mathbf{q}$, we can then measure the distance $\mathcal{D}$ between its geographical click probability distribution $\mathbf{q}$ and the user population distribution $\mathbf{p}$ approxi-

mated by Equation 6.4:

$$\mathcal{D} = \mathbf{dist}(\mathbf{p}, \mathbf{q}) \tag{6.5}$$

where **dist** can be an appropriate standard distance measure of choice, e.g., *chi-squared* χ^2, *KL Divergence*. Thus, given a query $\mathbf{q}$, we can classify $\mathbf{q}$ as follows:

$$\begin{aligned} \mathbf{q} &\in GSQ, \mathcal{D} > \tau \\ &\in NGSQ, \text{otherwise} \end{aligned}$$

where τ can be a distance value trained using a training set of labelled *NGSQs*

Maximum-likelihood Model. The spreading factor α is an indicator of how *local* a query or a Web page is: a larger α indicates a query or a Web page of more *local* interests. Thus, given a labelled training set of $NGSQs = \{\mathbf{q_i'}\}$ and the corresponding spreading factors $\{\alpha_i\}$, we can calculate a function

$$\mathcal{N}(\{\alpha_i\}) = \widetilde{\alpha}, \tag{6.6}$$

where $\widetilde{\alpha}$ is the *norm* of the α values for such queries, for example, $\mathcal{N}(\{\alpha_i\}) = \sum_i \alpha_i / |\{\alpha_i\}|$.

Thus, we can classify whether a given query is a *GSQ* as follows:

$$\begin{aligned} \mathbf{q} &\in GSQ, \alpha > \widetilde{\alpha} \\ &\in NGSQ, \text{otherwise.} \end{aligned}$$

6.5 Improve relevance ranking for geo-sensitive queries

We discussed how to algorithmically identify geo-sensitive queries in the previous section. In this section, we propose to improve the relevance ranking for the geo-sensitive queries by taking into account the geographical interests of the users

implicitly embedded in the search queries.

We see in Section 6.2 that for a search query, the patterns of its geographical click distribution indicates the locality preferences of the users. For example, the query *"amc mercado 20"* receives more user clicks from Santa Clara, CA than any other area, which may indicate that this query about a local movie theater is indeed more *relevant* to search users in the same geographical region. Thus, the click map of this query will likely show high density of clicks in the Santa Clara region. For the same reason, Web pages that are relevant to the AMC Mercado 20 movie theater will more likely register clicks from local search users than users from other locations (e.g., search users in New York City). Thus, the geographical distribution pattern of the aggregated user clicks on these pages may also show a concentration of clicks in the Santa Clara area. This observation suggests we study the correlation between the geographical distribution of interests of the query and of the corresponding Web pages as a measure of their *relevance*.

Vector Model. We denote a search query q by a k-dimension probability distribution $\mathbf{q} = \langle d_1, d_2, \cdots, d_k \rangle$, and denote a Web page w by $\mathbf{w} = \langle d_1^w, d_2^w, \cdots, d_k^w \rangle$, where each dimension d_i or d_i^w represents *the probability of registering user clicks from a geographical region* r_i. Thus, we can measure the *similarity* in geographical click distributions of the query and the Web page as the *cosine similarity* of the two vectors:

$$\mathcal{S}(\mathbf{q}, \mathbf{w}) = \frac{\mathbf{q} \cdot \mathbf{w}}{\|\mathbf{q}\| \|\mathbf{w}\|} \tag{6.7}$$

where $\cdot$ denotes the dot product operation and $\| \cdot \|$ denotes the magnitude.

Alternatively, it can also be measured by the probability for the Web page $\mathbf{w}$ to be clicked by users who issue the query $\mathbf{q}$, given their corresponding probability distributions,

$$\begin{aligned} P_c(\mathbf{w}|\mathbf{q}) &= \sum P_r(\mathbf{w}|\mathbf{q}) \\ &= \sum \frac{P_r(\mathbf{q}|\mathbf{w}) \cdot P_r(\mathbf{w})}{P_r(\mathbf{q})} \end{aligned}$$

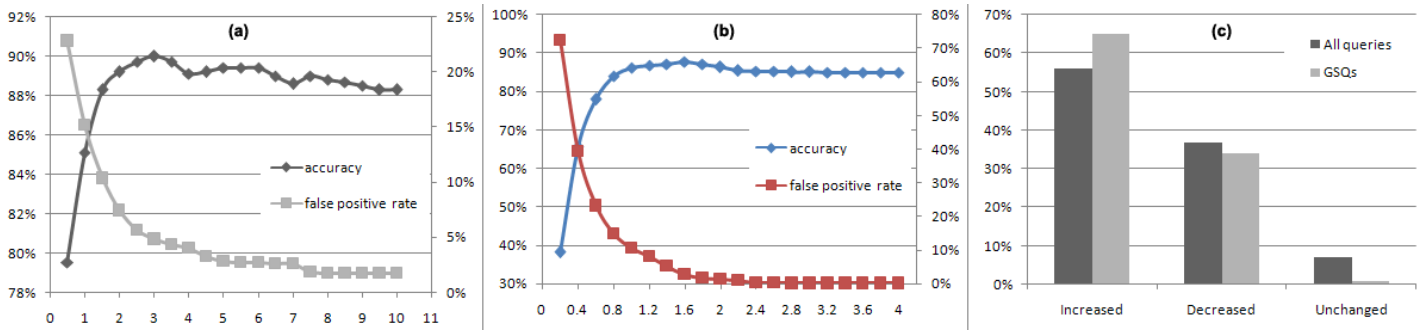

Figure 6.2. (a): Classification accuracy and false positive rate of the vector model classifier. (b): Classification accuracy and false positive rate of the maximum-likelihood classifier. (c): Percentage of all queries and *GSQ*s with increased, decreased, and unchanged DCG scores. It is not surprising to see that there is more relevance improvement when we apply the model on a set of queries comprised entirely of *GSQ*s.

where $P_r(\mathbf{w}|\mathbf{q})$ denotes the probability of a Web page $\mathbf{w}$ receiving user clicks from a geographical region r given a query $\mathbf{q}$, $P_r(\mathbf{q}|\mathbf{w})$ denotes the probability of a query receiving user clicks from a region r given the Web page $\mathbf{w}$, $P_r(\mathbf{w})$ denotes the probability of a Web page $\mathbf{w}$ receiving user clicks regardless of the search query, and $P_r(\mathbf{q})$ denotes the probability of a Web page $\mathbf{q}$ receiving user clicks regardless of the returned Web page.

Maximum-likelihood Model. Given a query $\mathbf{q}$ and a Web page $\mathbf{w}$, let $p_q = C_q \cdot d_q^{-\alpha_q}$ and $p_w = C_w \cdot d_w^{-\alpha_w}$ denote the click probability at a point at distance d_q and d_w away from the centers of interests $\mathcal{C}_q$ and $\mathcal{C}_w$, correspondingly. We can measure the proximity of two centers of interests w.r.t. the corresponding spreading factors as

$$\mathbf{P}(\mathbf{q}, \mathbf{w}) = C_q d_c^{-\alpha_q} \cdot C_w d_c^{-\alpha_w}, \tag{6.8}$$

where d_c is the distance between $\mathcal{C}_q$ and $\mathcal{C}_w$. Intuitively, $\mathbf{P}(\mathbf{q}, \mathbf{w})$ becomes maximal when $d_c = 0$, i.e., when the two centers of interests converge.

6.6 Experimentation and Discussions

6.6.1 Query Geo-sensitivity Classification

To establish the ground-truth for evaluating the classification accuracy, a test-set of 1,000 queries were randomly sampled from the search logs, and each query in the test-set was classified by a team of professional editors according to the *geo-sensitivity definitions* outlined in Table 6.1.

We limited the scale of experiments to geographical locations within the continental United States, and fixed the region granularity as a State. Thus, $\mathbf{q}$ is a 48-dimension vector, and each dimension represents the probability of user clicks from the corresponding State. Same for $\mathbf{p}$. We chose χ^2 as the distance metric.

label	**vector model**	**maximum-likelihood model**
GSQ	*metro community college, utah power, dixon middle school, kcci, idaho lottery, alaska weather, salt lake tribune newspaper, oklahoma county assessor, SLED, job openings in alabama, utah jazz, CCSU, iowa dmv, oregonian, green bay press gazette*	*MJR Waterford Cinema 16 Waterford MI, whittemore speedway, mercedes benz san antonio, berkeley nj, job openings in alabama, speaker of baseball, deaf smith county, idaho lottery, maui vacation rental, city of brawley, south plains college, wtvm, utah power, kcci, puerto vallarta hotel*
NGSQ	*online dictionary, internet explorer, adobe, winzip, famous quotes, microsoft word, the notebook, ipod shuffle, mcafee, spanish translator, xbox, us postal service, poker, matrix, colors*	*wwe, elton john, meditation, geri halliwell, salary, taxi, charlie sheen, supreme court, Uma Thurman, eliza dushku, toledo spain, torrent search, the producers, baseball, mcafee*

Table 6.2. Top 15 queries classified as *geo-sensitive* or *non-sensitive* by the vector model and the maximum-likelihood model.

We first calculated $\mathbf{p}$ using the 10K queries training dataset, and then applied the proposed classification methods on the 1K queries test set. Evaluated against the ground-truth editorial labels, the two models showed quite promising accuracy. In Figure 6.2(a) and 6.2(b), the X-axis represents $\mathcal{D}$ for the vector model and α for the maximum-likelihood model; the Y-axis on the left applies to the accuracy curve, and the Y-axis on the right applies to the false positive rate curve. Here

the *classification accuracy* is the ratio of queries that were correctly classified as either *GSQ* or *NGSQ*, and the *false positive rate* is the ratio of *NGSQ*s that were incorrectly labeled as *GSQ*s. The vector model achieved 90% accuracy and 4.8% false positive rate when $\mathcal{D} = 3.0$ (see Table 6.3 for the confusion matrix), while the maximum-likelihood model achieved 87.8% accuracy and 2.8% false positive rate when $\alpha = 1.6$. For most queries, the vector model was an order of magnitude faster to compute than the maximum-likelihood model.

	GSQ	***NGSQ***
Labeled as *GSQ*	91	41
Labeled as *NGSQ*	59	809

Table 6.3. The confusion matrix of the classification results using the vector model with $\mathcal{D} = 3.0$.

Table 6.2 shows the top 15 queries that were classified as either *GSQ* or *NGSQ*, order by $\mathcal{D}$. There are several *GSQ*s that are worth mentioning. For example, *"kcci"* is a local TV station in Des Moines, IA. *"SLED"* is the acronym for the *South Carolina Law Enforcement Division*. *"CCSU"* is the shorthand for *Central Connecticut State University*. And *"wtvm"* is a television station in Columbus, Georgia. All these queries would probably not have been easily identified as *geographically sensitive* had we relied only on the syntactic information of the queries. It is also worth noting that the top queries identified using the two models did not have much overlap. In particular, the maximum-likelihood model picked up much more celebrities' names as *NGSQ*s (five out of the top 15) than the vector model.

6.6.2 Relevance Improvement for Geo-Sensitive Queries

To measure the improvements w.r.t. search relevance ranking, we used a popular metric Discounted Cumulative Gain (DCG) [59, 61] as the evaluation metric. DCG allows us to attach more importance to documents that are ranked higher in the list, and to differentiate various levels of subjective relevance judgment by human evaluators. For a given query q, DCG is defined as

$$DCG(q, N) = \sum_{d=1}^{N} \frac{2^{R(d)} - 1}{\ln(1 + d)} \tag{6.9}$$

where $R(d)$ is the editorial rating of the $d-$th Web page in the top N ranked search results. Intuitively, a higher DCG score indicates a better relevance ranking.

We evaluated the relevance improvement on approximately 8,900 search queries sampled from the query logs of Yahoo! Search. For each of these queries, we calculated the DCG scores for a baseline ranking algorithm, and an alternative ranking algorithm that took into account the geographical distribution of user clicks. The baseline ranking algorithm was a proprietary decision-tree-based ranking algorithm trained on a very large number of features, and generated the same search ranking as Yahoo! Search. On the other hand, the alternative ranking algorithm utilized $\mathbf{D}(\mathbf{q}, \mathbf{w})$ in Equation 6.7 as one important ranking feature, in addition to the existing features being used in the baseline algorithm. We then compared the DCG scores of the alternative algorithm to the baseline.

On average, the alternative ranking algorithm which utilized the features generated by our click-based model outperformed the baseline algorithm by 1% at a significance level of $\alpha = 0.05$. This improvement was achieved on all queries regardless of whether they were *GSQs* or *NGSQs*, while existing work typically measure such improvement on a manually labelled *GSQs* only. Comparing the percentage of queries with improved and degraded DCG scores in *GSQ*∪*NGSQ* and in *GSQ*, the percentage of improved was increased by 16%, while the percentage of degraded was decreased by 8%. See Figure 6.2(c) for a breakdown by query class and Table 6.4 for a sample of the geo-sensitive queries with increased DCG scores.

query	**DCG@1**	**DCG@10**
greeks pizza muncie	+2.7%	+3.7%
st louis massage school	+1.0%	+13.9%
el paso texas flooding	+ 2.2%	+27.1%

Table 6.4. Examples of geo-sensitive queries with DCG improvement.

6.7 Disambiguating and Clustering Search Results

We briefly discuss some preliminary findings in using geographical click distribution to cluster and disambiguate search results for ambiguous queries. We report our experiments on a query *"washington jobs"*, which is ambiguous in terms of the user's geographical preference: is it about jobs in the Washington state or Washington DC?

Our goal was to cluster the search results such that Web pages in the same cluster were relevant w.r.t. the same geographical region. First, we aggregated past user clicks per URL, and modeled each click as a vertex on a two-dimension Euclidean space, denoted by (*latitude*, *longitude*) in the geographical space from where this click came from. Then we constructed an N nearest-neighbor graph G and repeatedly split G into K-clusters using the *Min-Max Cut* partitioning algorithm. After the vertexes (clicks) were all clustered, we clustered a given Web page based on its likelihood of receiving clicks in each of the click clusters. Eventually, each Web page in the search results was represented by a probability distribution over the K clusters.

Figure 6.3 visualizes the result of the document clustering process. Each dot represents a Web page returned by the search engine, and different colors denote different clusters. Web pages relevant to different geographical regions are mostly separated, and those that are relevant to the same region are grouped together into the same cluster.

6.8 Summary

In this chapter, we studied two click-based models of geographical distribution of users' interests. Based on the implicit user feedback in the form of user clicks, we can indirectly model users' locality preference. We then discussed two applications of the click-based models. We first applied the models to classify search queries based on their locality preference, and also identify users' *distribution of interests* for such queries. We then adapt the models to generate meaningful metrics for measuring the proximity between a search query and a Web page w.r.t. locality

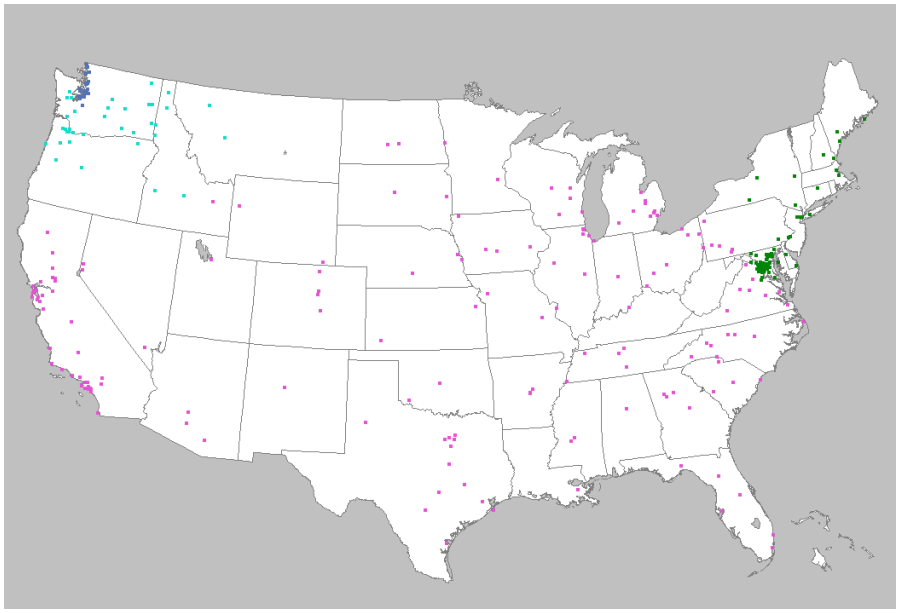

Figure 6.3. Visualization of search results clustering for ambiguous query *"washington jobs"*. Dots in different colors denote different clusters, e.g., the green dots denote Web pages relevant to Washington DC, and the blue dots relevant to the State of Washington.

preference, with the goal to improve search ranking relevance. We presented the results from our empirical experiment on a large dataset, and the two models demonstrated promising results.

Chapter 7

Mining Program Committee Characteristics for Venue Recommendation

In a way similar to a relevance ranking system, a venue recommendation system measures the quality of the publication venues in a library, ranks the venues, and recommends relevant and high-quality venues according to the ranking. Traditionally, bibliometrics have been an important measure for venue quality in digital libraries. When a user searches through literature or a librarian makes a subscription decision, bibliometrics are used to measure the quality and impact of venues and documents.

In this chapter, we study how *internal feedback* can be leveraged for venue recommendation in digital libraries. Specifically, we investigate the feedback mechanism between a venue and its organizers, illustrated in Figure 7.1.

To make the problem more tractable, we tackle the scenario in which digital libraries in the Computer Science domain, such as the ACM Portal[1] and CiteSeer digital library[2], need to automatically measure the quality of academic conferences. This is a non-trivial problem for two reasons. First, the Computer Science discipline is unique in its publication practice: unlike almost every other field, peer-reviewed conferences play a role equally (if not more) important to that of

[1] http://portal.acm.org/dl.cfm
[2] http://citeseer.ist.psu.edu/

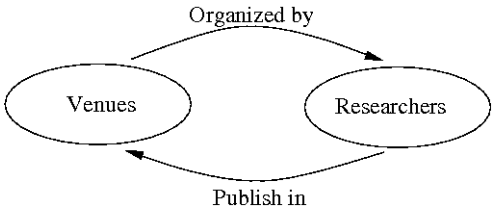

Figure 7.1. A two-way feedback loop: publication venues are typically organized by a group of researchers, and researchers write papers to publish in the venues. How does the quality of the venues and the characteristics of the organizing researchers impact each other?

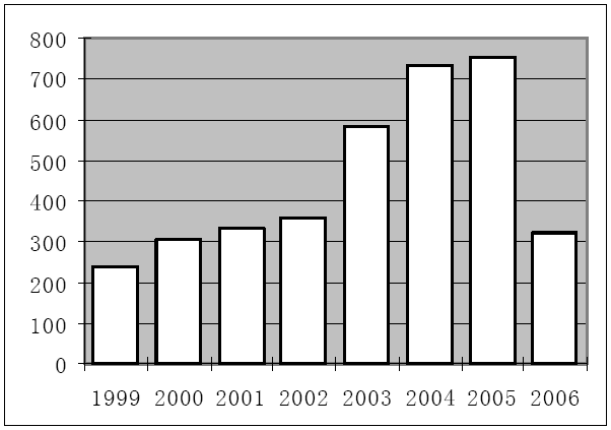

Figure 7.2. Number of distinct conference Call for Papers announced on DBWorld in each year from 1999 until May 2006.

the established journals. The discipline's fast-moving pace of progress requires that new research findings to be distributed more quickly and on a broader scale. With competitive acceptance rates of 10-20% and often receiving more citations than journals [99], prestigious refereed conferences are one of the premium publishing venues for researchers in Computer Science.

The fast-moving pace of progress in the discipline of Computer Science requires that new research findings are distributed more quickly and on a broader scale. As a result, the discipline is unique in its publication practice: unlike almost every other field, peer-reviewed conferences play a role equally (if not more) important to that of the established journals. With competitive acceptance rates of 10-20% and often receiving more citations than journals [99], prestigious refereed conferences are one of the premium publishing venues for researchers in Computer Science.

Second, with the rapid growth of the Computer Science discipline, the number

of conferences has also increased dramatically in recent years. Figure 7.2 shows the number of event announcements published to DBWorld[3] between 1999 and 2006. Often confronted with an abundance of available venues, it is becoming more and more important for researchers and librarians to be discerning about the reputation (thus the quality) of the conferences.

We study the characteristics of the conference program committee members, and investigate whether they are good indicators of the quality of the conference. Formally, the problem can be defined as follows:

Conference Quality Measure: Given a set of conferences $\mathcal{V}$ and the program committee, with multi-attribute information such as {size of committee, publication records of committee, affiliations of committee, ...}, identify a set of reputable or low-quality conferences $\mathcal{R} \subseteq \mathcal{V}$, such that any conference $\mathbf{v_r} \in \mathcal{R}$ satisfies the constraint ξ (ξ to be given).

7.1 Background and Motivation

Existing techniques to measure the reputation and quality of venues typically associate conferences with certain bibliometrics, most notably citation-based metrics such as the Impact Factor [49]. That is, the quality of a venue is in direct proportion to its bibliometrics characteristics. For example, if the number of citations to the publications in a conference has passed a certain threshold, the conference is considered of good quality (to be elaborated in Section 2). However, in this paper, we claim that such techniques are inadequate to measure the quality of conferences in Computer Science:

1. For emergent or young conferences, historical citation statistics are not readily available, rendering citation-based metrics inapplicable.

2. Even for well-established conferences, citation statistics takes time to accumulate. A recent study of the major database conferences and journals between 1994 and 2003 shows that many of the citations reach back five and more years [99].

[3] http://www.cs.wisc.edu/dbworld/

When we read the *Call for Papers* (CFP) of a conference and try to decide whether it is a reputable venue worth publishing in, we usually examine the list of Program Committee (PC) members and make our reasonable judgment. Thus, we hypothesize the following:

Hypothesis 6.1 *The characteristics of the PC members correlate with the quality of a conference they are organizing.*

This hypothesis and the aforementioned two issues have inspired us to study the problem of venue ranking and recommendation from a novel perspective – to measure the quality of conferences through analyzing the characteristics of their PC members.

In this paper, we explore an array of heuristics for mining the feedback loop between the characteristics of the PC members and the quality of the conferences. Given a large collection of conference CFPs, we first employ entity extraction techniques to recognize and extract the names of the PC members. By mining their characteristics, our proposal allows the automatic measuring of venue quality, even when the publication citation statistics are scarce or unavailable, thus the aforementioned issues can be resolved.

For digital libraries, there are several directions toward which this work can be applied. First of all, we provide a novel method of estimating the impact of conferences, and it can be fully automated. This method can filter through the CFPs of emerging conferences, spotting possibly prestigious ones to recommend to researchers and librarians. Second, the proposed heuristics can be applied to approximate the existing citation-based conference impact factors, especially when the citation records of this conference are scarce or inaccessible.

The rest of this chapter is organized as follows. In Section 7.2, we describe our experimental framework and the real-world data sets which we have collected. In Section 7.3, we propose five heuristics to identify reputable conferences through PC characteristics analysis, and investigate each of them individually. In Section 7.4, we combine these heuristics in a classification scheme and examine how well they work in aggregation. We report the performance of our classification algorithm in detecting a set of low-quality conferences in Section 7.5. In Section 7.6, we

extend the proposed heuristics for conference ranking. In Section 7.7, we discuss a number of implications based on the experimental results. We summarize our work in Section 7.8. In Section 7.9, we present an online prototype system for rating the quality of conferences.

7.2 Data Collection

We used two datasets in the design and evaluation of our algorithm. First, we used the citation data from the ACM Guide covering a 54-year range from 1950 to 2004, which contained the metadata about 609,000 authors and 770,000 articles. The ACM Guide is a high-quality citation digital library that has a good coverage on the computing literature. We used the data to construct a collaboration graph [90], in which nodes represent authors and edges between any two nodes represent coauthorship (i.e., two authors have coauthored one or more papers). All edges in the graph are unweighted, that is, all have the equal importance and only signify whether a collaboration exists between two authors. Repeated collaborations between two authors can also be captured into the graph by weighting each edge with a value proportional to the number of publications two authors have coauthored. Such a graph with weighted edges may give more hints about the collaboration strengths among the authors. The ACM Guide has generated about 1.2 million edges in our collaboration graph. Note that ACM Guide itself does not have a notion of "unique key" such as Digital Object Identifier (DOI). Instead, it depends on the names of authors to distinguish them. Therefore, the classical name authority control problem [56] may arise (i.e., same author with various spellings or different authors with the same spelling). We carefully tried to minimize the impact of this problem and in experiments always used the full names whenever possible (e.g., "Dongwon Lee" instead of "D. Lee").

We collected 2,979 unique conference CFPs between February and May 2006 from DBWorld, a comprehensive and frequently-updated list of events in Computer Science (with the focus on database-related topics). We denote this conference dataset as $\mathcal{V}$. Note that throughout the rest of the chapter, we shall use the term "*conferences*" to represent conferences, workshops, and symposiums, as our proposal would work regardless of the type of event, as long as CFPs are given. We

Topic	Conferences
Database	SIGMOD, VLDB, ICDE, EDBT, ...
AI	AAAI, IJCAI, ICML, NIPS, KDD, ...
Application	WWW, ICCV, ACL, ...
System	HPCA, CCS, ASPLOS, DAC, ...
Theory	STOC, SIAM, FOCS, LICS, SCG, ...

Table 7.1. Examples of reputable conferences $\mathcal{R}$.

then extracted 16,147 what we believed to be distinct PC members from the CFPs with the same disambiguation techniques applied on the ACM dataset. Using one's first and last names as the mapping key, these PC members were matched with the ACM dataset. About 75.63% of the PC members had a 1:1 mapping to the ACM dataset.

Next, based on the conference impact factor ranking from CS ConferenceRanking.org[4], we extracted the top 20 ranked conferences in each sub-field of Computer Science, and obtained their authoritative full names from DBLP[5]. Then we extracted from the DBWorld dataset 576 CFPs that approximately match the names of these top conferences. The resulting 576 CFPs were labeled as $\mathcal{R}$, which formed a representative training set for the CPFs of the *reputable* conferences (see Table 7.1). The remaining conferences $\mathcal{V} \setminus \mathcal{R}$ were labeled as $\mathcal{C}$, which contained 2,403 CFPs and was disjoint from $\mathcal{R}$. At the end, $\mathcal{R}$ consisted of about 19.34% of all the CFP data collected, a reasonable sample of the top 20 ranked conferences.

7.3 Identify Reputable Conferences

In this section, we explore a number of heuristics with the focus on the higher end of the quality spectrum. The remainder of this section discusses the proposed heuristics in detail, and investigates the effectiveness of each heuristic as shown in evaluations performed on these two datasets $\mathcal{R}$ and $\mathcal{C}$.

[4]http://www.cs-conference-ranking.org/
[5]http://www.informatik.uni-trier.de/~ley/db/

7.3.1 Number of PC members

Typically, a good conference reaches a large audience, receives a significant amount of submissions due to its popularity, and as a result requires a large program committee to facilitate the rigorous review process.

In our first experiment, we investigated whether the number of PC members has a strong correlation with the quality of a conference. We plotted the distribution of the number of PC members in our dataset in Figure 7.3. This figure (and all the other figures in Section 4) consists of two sub-figures (style adopted from [91]). The sub-figure on top Figure 7.3(a) is a histogram showing the differences in the frequency distribution of the two types of instances, $\mathcal{C}$ and $\mathcal{R}$. The sub-figure at the bottom Figure 7.3(b) consists of a bar chart and a line chart. The X-axis represents a heuristic under investigation (currently showing *the number of PC members*). The Y-axis on the left, which applies to the bar chart, depicts the overall percentage of the 2,979 conferences that fall into a particular range of the current heuristic. The Y-axis on the right, which applies to the line chart, depicts the probability of the conferences within that particular range that are labeled as $\mathcal{R}$ (i.e., judged as reputable conferences). Here the probability is calculated as the percentage of instances labeled as $\mathcal{R}$.

Overall, Figure 7.3 shows that the number of PC members tends to be larger in $\mathcal{R}$ than in $\mathcal{C}$: the mean of $\mathcal{R}$ is 37.28 compared with 26.83 in $\mathcal{C}$. Although this heuristic exhibits a clear correlation, the chart becomes noisy and shows a number of spikes toward the right, most possibly due to the small number of data points within range.

7.3.2 Average number of publications by PC

Typically, a reputable conference has a program committee of renowned researchers. Because one's number of publications is usually an indicator for the quality of his/her research, we studied in the second experiment whether the average number of publications of the PC members (data collected from the ACM dataset) is a good indicator for the quality of the conference. The results are shown in Figure 7.4. Although PC members of the conferences in $\mathcal{R}$ appear to be more prolific, the difference is not highly significant between $\mathcal{R}$ and $\mathcal{C}$: mean of $\mathcal{R}$ is 16.94 and $\mathcal{C}$ is

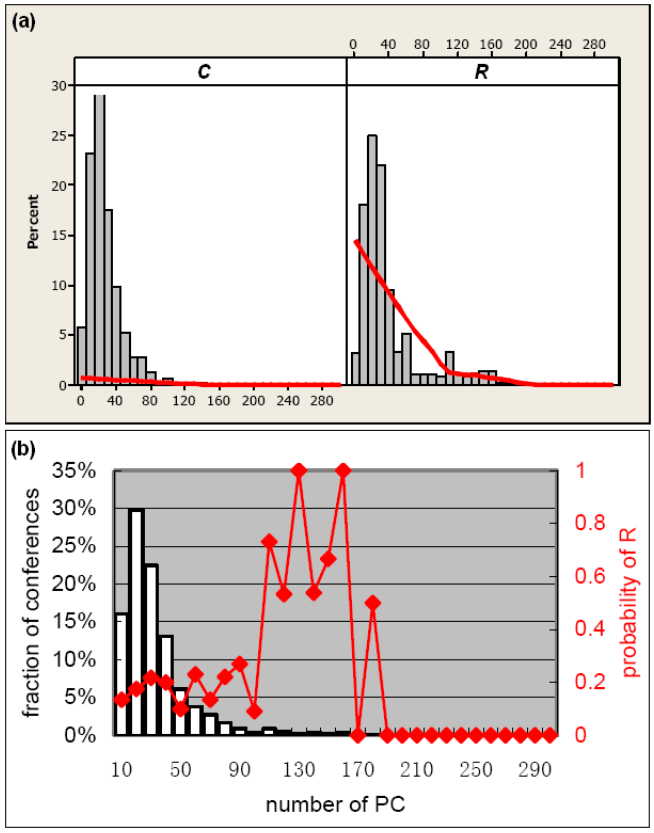

Figure 7.3. (a) Conferences labeled as $\mathcal{R}$ tend to have more PC members, which coincides with (b), showing the prevalence of $\mathcal{R}$ is higher among conferences with more PC members.

13.44. The prevalence of $\mathcal{R}$ actually drops as the average number of publications increases beyond about 40, most likely due to the scarcity of data.

7.3.3 Average number of coauthors of PC

A related heuristic is to observe how frequently the PC collaborate with their peers in the field. The assumption is that renowned researchers tend to be more active in collaboration with their colleagues for publication. Thus we investigated whether the average number of coauthors of PC members has a positive correlation with the conference quality. Note that number of coauthors specifies the number of *distinct* collaborators on all publications of a given author. Figure 7.5 depicts that, generally speaking, a larger number of collaborators of the PC members implies a higher chance of the conference being a reputable one. The mean for $\mathcal{R}$ is 16.34

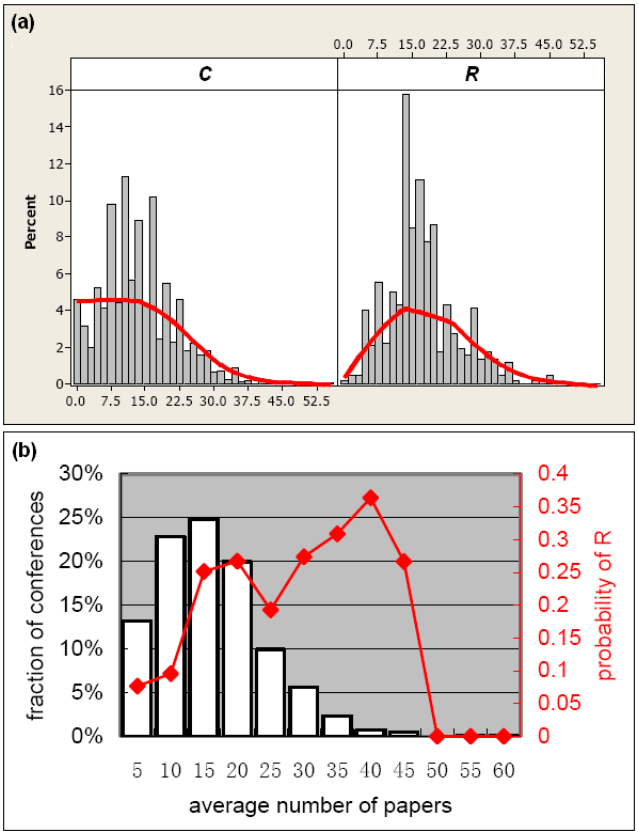

Figure 7.4. PC members of conferences labeled as R tend to have more publications than their counterparts in C, however the difference is not very significant.

and for $\mathcal{C}$ is 13.09. However, using this heuristic alone may lead to false positives, as some prominent researchers mainly publish single-authored papers. But this is a diminishing trend in computer science as the average number of collaborators per author tends to increase steadily (Figure 7.6). Therefore we expect a low false positive rate.

7.3.4 Closeness centrality of PC

The *closeness centrality* measure is one of the methods in Social Network Analysis (SNA) to quantify an individual's location in a community [127]. Most prominent ones are often located in the strategic locations in the social network of the community. The *closeness centrality* measure can be defined as how close an author is on average to all other authors. Then, authors with high *closeness* values could be viewed as those who can access new information more quickly than others, and

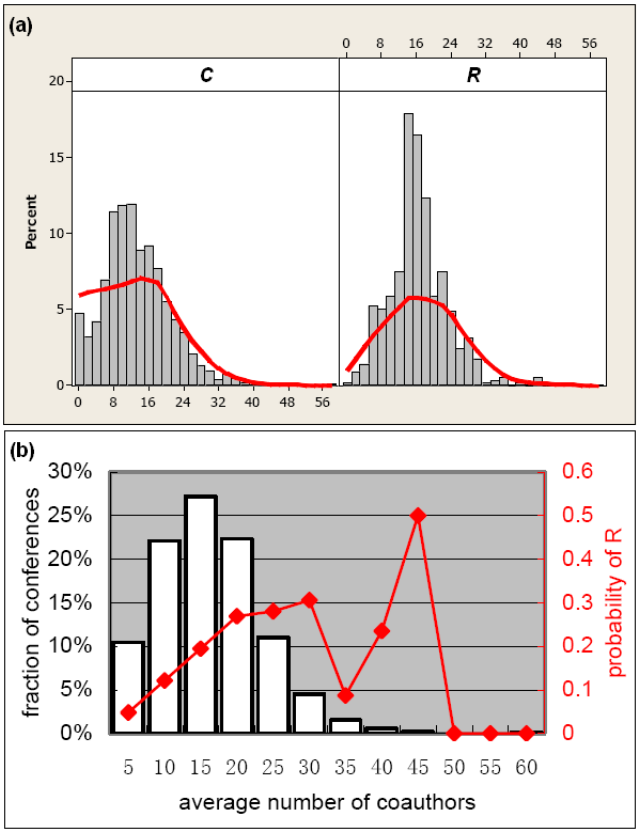

Figure 7.5. The distribution is reasonably similar to that of 4.2. PC members of conferences labeled as R tend to have more collaborators.

similarly, information originating from those authors can be disseminated to others faster [90]. Formally, the closeness of a node v in a connected graph G is defined as follows:

$$CC(v) = \frac{n-1}{\sum_{w \in G} d(v,w)}$$

where $d(v,w)$ is the pair-wise geodesic (i.e., shortest distance) and n is the number of all nodes reachable from v in G.

In our fourth experiment, we studied whether the average *closeness* of the PC members has a correlation with the quality of the conference. Here we calculated the *closeness* value for every PC member based on the collaboration graph constructed using the ACM dataset. The assumption is that a high quality conference has a program committee composed of a group of renowned researchers, who are prominently located in the social network of the community.

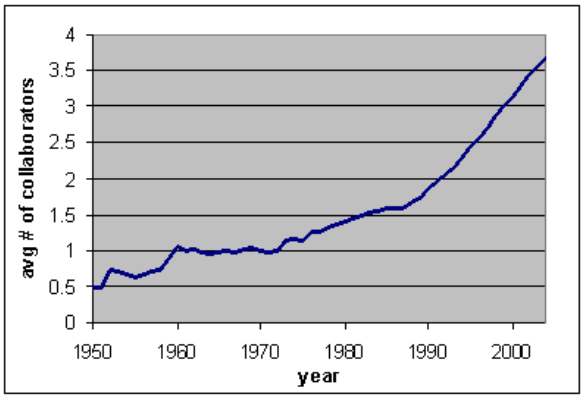

Figure 7.6. The average number of distinct collaborators per author in the ACM dataset is steadily increasing over time.

As can be observed in Figure 7.7, the prevalence of reputable conferences is generally on the rise as the average *closeness* of the PC members increases, with a spike toward the right when the average *closeness* reaches 0.1. The mean of $\mathcal{C}$ is 0.056 compared with 0.062 of $\mathcal{R}$.

7.3.5 Betweenness centrality of PC

While the *closeness centrality* measure depicts how visible an individual is in the community, the *betweenness centrality* measure shows how influential an individual is over the information flows in the social network. Sometimes the interactions between any two indirectly connected authors (i.e., they never collaborated with each other before) might depend on the other authors who connect them through their shortest path(s). These authors potentially play an important role in the network by controlling the flow of interactions. Hence the authors who lie on most of the shortest paths between pairs of authors can be viewed as the *hubs of collaboration* in the community. This notion, known as the *betweenness* of a node v, $B(v)$, measures the number of shortest paths between pairs of nodes passing through v, and formally defined as follows [44]:

$$BC(v) = \sum_{w,x \in G} \frac{d(w,x;v)}{d(w,x)}$$

where $d(w,x)$ is the shortest path between w and x, and $d(w,x;v)$ is the shortest path between w and x passing through v. The equation can also be interpreted as

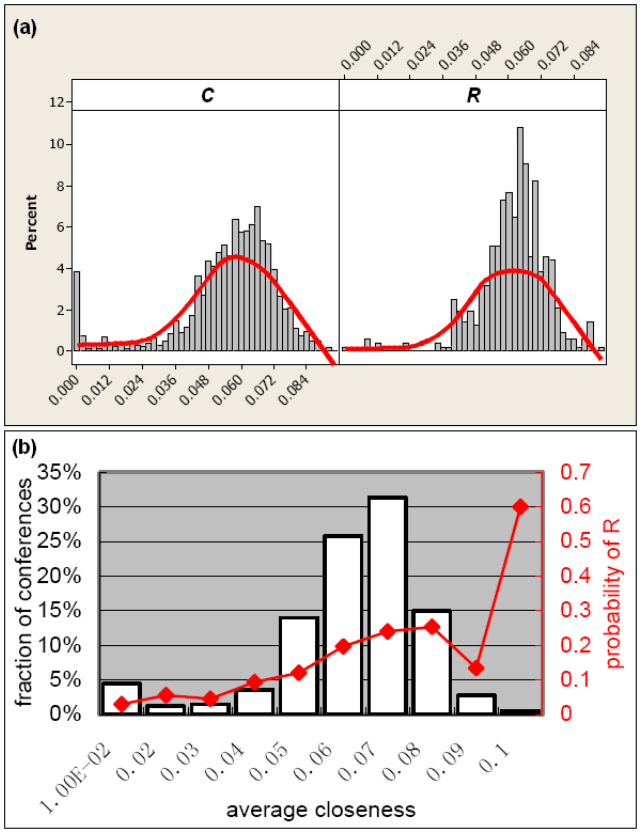

Figure 7.7. (a) The distribution between C and R shows some differences, and (b) it exhibits a positive correlation between the average *closeness* and the prevalence of R.

the sum of all probabilities that a shortest path between each pair of nodes w and x passes through node v.

Figure 7.8 depicts the correlation between the average *betweenness* of the PC members and the quality of the conference. We again calculated the *betweenness* value for each PC member based on the collaboration graph constructed using the ACM dataset. To our surprise, it appears that there is no strong correlation between the two. A spike exists when the average *betweenness* approaches 0.0003, however there are only five instances within that range.

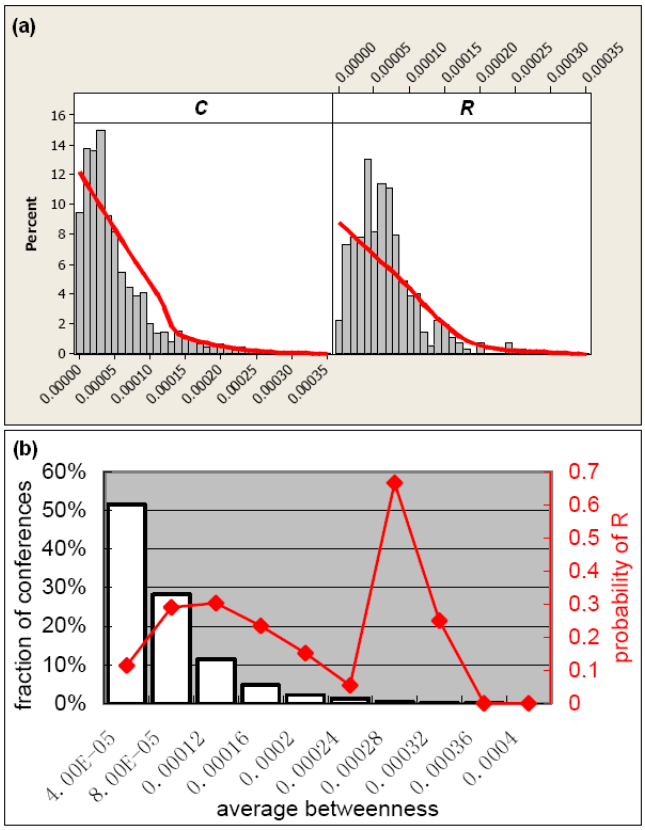

Figure 7.8. (a) The average *betweenness* of PC members in R is higher than $\mathcal{C}$, however (b) shows no clear correlation between this factor and the quality of the conference.

7.4 Combining Heuristics for Classification

Previously, we have presented a number of heuristics to identify reputable conferences through statistical analysis of the characteristics of the associated PC members. Although most of the characteristics exhibited various degrees of correlation with the quality of the conference, few of them have enough distinguishing power when used individually. For example, using the "number of PC members" as heuristic, the prevalence of R was considerably high within the range of 110 - 170, however only less than 5% of the conferences in our dataset fell within that range. Therefore in this section, we study how to combine these heuristics to determine high-quality conferences more efficiently and accurately.

7.4.1 Naive classification.

Without loss of generality, our problem in Definition 1 can also be cast to a *binary classification* problem, formally described as follows:

Conference classification: Given a set of conferences $\mathcal{V} = \{\mathbf{v}\}$, classify $\mathbf{v}$ into one of the two classes: (1) a class of reputable conferences $\mathcal{R}$ ($\mathcal{R} \subseteq \mathcal{V}$), and (2) a class of common or low-quality conferences $\mathcal{C}$ ($\mathcal{C} = \mathcal{V} \setminus \mathcal{R}$).

We report the classification results, using a C4.5 decision-tree classifier [95] with all the five heuristics described in Section 7.3 as the feature space.

We employed the ten-fold stratified cross validation technique [73] to evaluate the classification accuracy. This technique randomly divided the judged data into 10 partitions of equal size, and performed 10 training/testing phases in which nine partitions were used for training and the remaining tenth partition was used for testing. Therefore, in each training/testing iteration, 2,681 instances were used in training, leaving out 298 instances for testing.

First, we classified the instances using each of the five heuristics individually. The number of PC members heuristic performed the best, correctly identifying 2,436 (81.77%) of the instances, and only misjudged 22 (0.91%) of the $\mathcal{C}$ instances to be $\mathcal{R}$. The *average closeness* heuristic came in second, with the ability to correctly classify 2,403 (80.66%) of the instances.

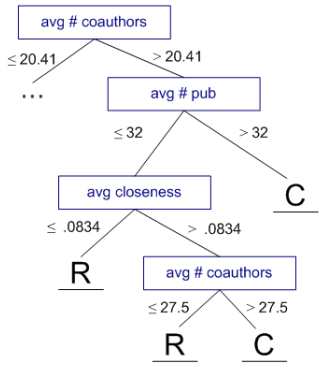

Figure 7.9. A portion of the induced C4.5 decision tree for the combined heuristics.

Then, we combined all the aforementioned heuristics under the C4.5 classifica-

tion scheme. After the ten-fold cross validation, the results were promising: 2,570 (86.27%) of the judged instances were classified correctly while 409 (13.73%) were classified incorrectly. The false positive rate for class $\mathcal{R}$ was low at 0.035. Table 7.2 shows a precision-recall matrix, in which the precision measure indicates the percentage of correctly classified instances in each class, and the recall measure indicates the fraction of instances that have been correctly classified. A portion of the resulting decision tree from the classification scheme is shown in Figure 7.9, and the full model is presented in Appendix B. In this tree, for example, a conference with an average number of coauthors of PC smaller than 20.41, number of PC smaller than 8, and an average number of publications of PC smaller than 20 is classified as $\mathcal{C}$. Table 7.3 shows the confusion matrix for the naive decision-tree classifier.

class	**precision**	**recall**
$\mathcal{C}$	0.877	0.965
$\mathcal{R}$	0.751	0.434

Table 7.2. Precision and recall for the naive classifier.

	$\mathcal{C}$	$\mathcal{R}$
Labeled as $\mathcal{C}$	2,320	326
Labeled as $\mathcal{R}$	83	250

Table 7.3. Confusion matrix for the naive classifier.

7.4.2 Boosting and bagging

Boosting [46] and bagging [96] are two of the most popular techniques for improving the accuracy of a given classification algorithm [96], and have been proven effective in detecting spam web pages [91]. The rationale of both techniques is to produce an accurate classification by combining the power of multiple less accurate classifiers that are trained iteratively. Bagging is similar to a *voting* process, in which the final outcome is the prediction made by the majority of the classifiers. And boosting goes further by assigning different weights to each of the predictions, iteratively forcing the classifier to learn more from their past mistakes.

In this section, we report the results from adopting these techniques to improve the performance of our classifier. A ten-fold cross validation was used in all the

experiments. Table 7.4 shows the precision and recall for the two classes $\mathcal{C}$ and $\mathcal{R}$, after applying bagging to the C4.5 classifier combining all the heuristics. After 10 iterations, both precision and recall were improved dramatically, especially for $\mathcal{R}$. Overall, the number of correctly classified instances was 2,663 (89.33%), increased from 2,570 (86.27%) before bagging. Table 7.6 shows the confusion matrix.

The boosting technique further improved the performance of our classifier. After another 10 iterations, we were able to correctly classify 2,739 (91.94%) instances, misjudging only 240 (8.06%). Table 7.5 shows the precision and recall after applying the boosting technique. Notice that the recall for $\mathcal{R}$ was again dramatically improved beyond bagging. 422 instances in $\mathcal{R}$ were correctly identified, increased from 311 with bagging. Table 7.7 shows the confusion matrix.

class	**precision**	**recall**
$\mathcal{C}$	0.899	0.979
$\mathcal{R}$	0.859	0.54

Table 7.4. Precision and recall after bagging.

class	**precision**	**recall**
$\mathcal{C}$	0.938	0.964
$\mathcal{R}$	0.831	0.733

Table 7.5. Precision and recall after boosting.

	$\mathcal{C}$	$\mathcal{R}$
Labeled as $\mathcal{C}$	2,352	265
Labeled as $\mathcal{R}$	51	311

Table 7.6. Confusion matrix for the classifier after bagging.

7.5 Detect Low-Quality Conferences

In April 2005, a group of MIT students played a prank[6] on the conference – "World Multi-Conference on Systemics, Cybernetics and Informatics (WMSCI)" – known

[6]Details about the MIT students' prank and the SCIgen tool to generate random research papers can be found at http://pdos.csail.mit.edu/scigen/

	$\mathcal{C}$	$\mathcal{R}$
Labeled as $\mathcal{C}$	2,317	154
Labeled as $\mathcal{R}$	86	422

Table 7.7. Confusion matrix for the classifier after boosting.

for sending unsolicited invitation emails to people in academia. The MIT students used software to generate bogus research papers, complete with context-free grammar, and submitted two of them to the conference. To their surprise, one of the gibberish papers was accepted without any reviews. The event received much attention, being covered in various media and has became an amusing topic for debate among scientists. Inspired by this happening, we continued our evaluation by investigating whether the proposed heuristics could be used to detect conferences on the other side of the reputation spectrum - the so-called *low-quality conferences*. We collected the CFPs of 18 low-quality conferences[7] by consulting colleagues and reading the online comments (or complaints) about certain conferences[8]. These conferences were labeled as $\mathcal{LQC}$.

We then tested each of the five heuristics on the two datasets $\mathcal{C}$ and $\mathcal{LQC}$. Overall, the differences were consistently obvious. In the interest of space, we report two most distinguishing heuristics in this section. It is apparent that on average the PC members in $\mathcal{C}$ are much more prolific than those in $\mathcal{LQC}$: the mean for $\mathcal{C}$ is 13.44 compared with merely 1.54 for $\mathcal{LQC}$ (see Figure 7.10).

On the other hand, the prevalence of $\mathcal{LQC}$ appeared to have a very strong correlation with the average closeness of the PC members (see Figure 7.11). The spike toward the left-hand side of the figure indicates that the lower the average closeness value, the more likely a conference belongs to the $\mathcal{LQC}$ class.

We also trained and tested a naive classifier to differentiate $\mathcal{C}$ and $\mathcal{LQC}$ using each and then all the aforementioned heuristics. When combined, the classifier correctly judged 2,408 (99.46%) of all the 2,421 instances. The precision and recall values for identifying $\mathcal{C}$ conferences were 0.996 and 0.998, and the false positive rate for $\mathcal{LQC}$ was very low at merely 0.002. See Table 7.8 for the classifier's confusion matrix.

[7]Identities of these conferences can be provided upon request.

[8]See http://www.inesc-id.pt/∼aml/trash.html and http://del.icio.us/tag/fakeconference

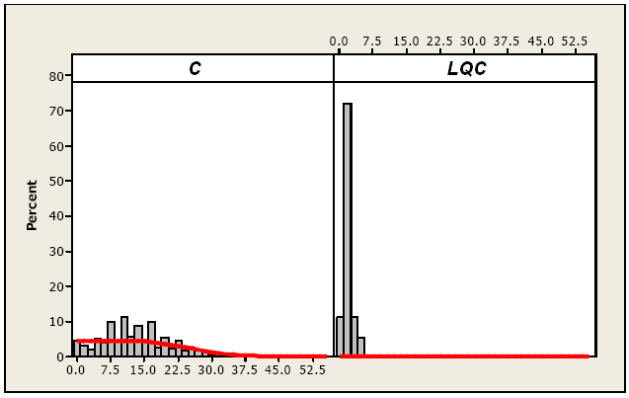

Figure 7.10. Overall the number of publications by PC members in $\mathcal{LQC}$ is significantly lower than that in $\mathcal{C}$; the highest average number of papers by PC members in $\mathcal{LQC}$ is only 3.8.

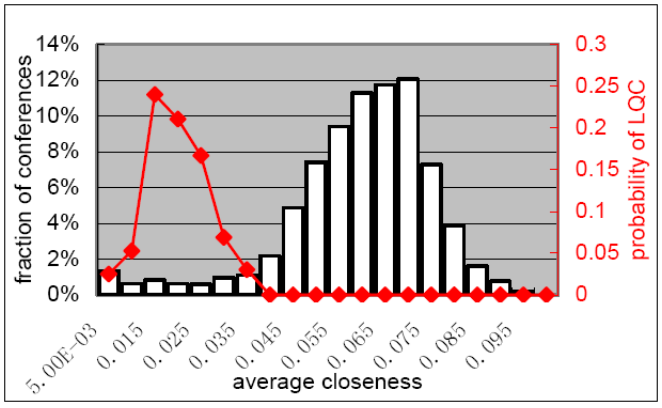

Figure 7.11. Conferences in $\mathcal{LQC}$ dominated the portion of conferences that had lower average closeness values.

7.6 Ranking Conferences

Many applications exist for digital libraries to automatically rank conferences based on their intrinsic quality. In inter- and multi-disciplinary academic units or in emerging topics such as bioinformatics, people with various backgrounds work together. However, there are numerous scenarios when measuring the quality of venues becomes important (e.g., making subscription decisions, recommending the most appropriate venue to submit work, promotion and tenure process, etc). What we have studied in previous sections is a supervised binary classification problem: to classify a given conference into either the reputable or the low-quality class.

In this section, we re-cast the problem into a ranking problem, applying our

	$\mathcal{C}$	$\mathcal{LQC}$
Labeled as $\mathcal{C}$	2,399	9
Labeled as $\mathcal{LQC}$	4	9

Table 7.8. Confusion matrix for the classifier to differentiate $\mathcal{C}$ and $\mathcal{LQC}$.

techniques to cover the whole quality spectrum:

Conference ranking: Rank a set of conference $\mathcal{V} = \{\mathbf{v}\}$ according to a scoring function $F(\mathbf{v}) = \nu, \nu \in [0, 1]$.

Furthermore, we make the following observations:

Observation 1: We can define the scoring function $F(\mathbf{v})$ as equivalent to the probability function $P(\mathbf{v} \in \mathcal{R})$, where $\mathcal{R}$ is a set of reputable conferences.

Observation 2: Let $\{\mathbf{v}_{\pi_1}, \mathbf{v}_{\pi_2}, \ldots, \mathbf{v}_{\pi_n}\}$ denote the permutation of $\mathcal{V}$ in the descending order of ν, then $\{\mathbf{v}_{\pi_1}, \ldots, \mathbf{v}_{\pi_k}\}$ and $\{\mathbf{v}_{\pi_{n-k}}, \ldots, \mathbf{v}_{\pi_n}\}$ are equivalent to the top K most reputable and the most low-quality conferences, respectively.

These two observations suggest a general ranking scheme to sort the conferences based on the likelihood of being reputable. We report two experiments in which we use a combination of all the aforementioned heuristics. In both experiments, we rank a set of conferences with unknown quality based on their probability of belonging to the set of hand-labeled reputable conferences $\mathcal{R}$ (previously described in Section 3 and used in Sections 4 and 5). Such probability is estimated by the C4.5 classifier [95] as the confidence of its prediction.

An overall ranking is reported by the first experiment. Not surprisingly, we see in Table 7.9 that most of the top-ranked conferences are overlapping with those that are present in $\mathcal{R}$.

More interesting results are reported in the second experiment, in which we only consider conferences that do ***not*** belong to $\mathcal{R}$. A motivation of this experiment is to investigate "how well the conferences in the middle region of the quality spectrum are ranked." Table 7.10 shows the top-10 conferences (that have not been labeled as $\mathcal{R}$) sorted by the prediction confidence. We closely examine these conferences by

FOCS 2004, ER 2004, ICDT 2005, MobiSys 2004, DEXA 2004, VLDB 2005, WWW 2004, ICDM 2004, SIGIR 2003, SIGMOD 2004, ACM SAC 2004, ICDE 2006 ...

Table 7.9. An example of the overall top-ranked conferences, which is highly overlapping with those in $\mathcal{R}$.

reading the conference websites and a random sample of the accepted papers, and consulting with our colleagues. Some are workshops co-located with prestigious conferences (e.g, IEEE INFOCOM '06, ICDE '06) and some are conferences that are ranked decently by the same source[9] from which we have collected our sample of reputable conferences (e.g., IDEAS is ranked 29th and ADBIS is 31st).

Another interesting observation of this ranking is that six out of ten are fairly recent venues in 2006[10]. This argues for the advantage of our ranking techniques – since these recent conferences most likely do not have citation statistics available, existing citation-based metrics such as the Impact Factor [49] become inapplicable. By exploiting the proposed heuristics, however, we are able to discover not only the *long-established* venues, but also the *emerging high-quality* ones.

7.7 Discussion

There are a number of ways in which we can improve the performance of our methods. For the task of detecting low-quality conference, a false positive judgment could be extremely detrimental, since we do not want to mistakenly label a legitimate and reasonably good conference as a low-quality one. Thus the goal is to have a high precision. Although the proposed heuristics scored well in terms of precision in our experiments, more representative training samples, especially of the low-quality conferences, should be collected in the next phase of evaluation. On the other hand, for the task of identifying reputable conferences, there are definitely rooms for improvements on the current recall value. In order to achieve better classification accuracy, several other heuristics can be utilized, e.g., both the affiliations and the number of accumulated citations of PC members can be

[9] CS Conference Ranking.org; see Section 3

[10] This experiment was done in late 2006.

rank	conference	prob.
1	2nd Intl. Conf. on Business Process Management (BPM) 2004	0.9268
2	11th Intl. Conf. on Multimedia Information Systems 2005	0.9268
3	4th Intl. Conf. on Business Process Management (BPM) 2006	0.9268
4	11th Intl. Conf. on Advanced Information Systems Engineering (CAiSE*99) 1999	0.9091
5	10th European Conf. on Advances in Databases and Information Systems (ADBIS) 2006	0.9091
6	Intl. Database Engineering and Applications Symposium (IDEAS) 1999	0.9000
7	9th IEEE Global Internet Symposium 2006	0.8889
8	4th Intl. Workshop on Adaptive Multimedia Retrieval 2006	0.8750
9	7th Workshop on Distributed Data and Structures (WDAS) 2006	0.8571
10	2nd IEEE Intl. Workshop on Networking Meets Databases(NetDB) 2006	0.7500

Table 7.10. Top 10 conferences that do *not* belong to $\mathcal{R}$. This shows how the ranking algorithm works in the middle region of the quality spectrum. Several recent venues (highlighted in rectangles) make their way into the list, which would not be possible in solely citation-based ranking scheme due to the lack of citation statistics.

good indicators for the general quality of the program committee.

The proposed heuristics rely on the completeness and correctness of the list of the PC members extracted from CFPs. One potential issue is that a small portion of CFPs do not have a complete list of PC members, e.g., only showing the committee chairs and organizers. In such cases, it requires further action to harvest the list of PC members (for example, by crawling the conference web sites) before the proposed heuristics can be applied.

7.8 Summary

In this chapter, we studied internal feedback between the quality of the conferences and the characteristics of the organizers, and leveraged the correlation to propose a

number of heuristics to identify reputable conferences for venue recommendation. When combined under a classification scheme, these heuristics performed promisingly, achieving a satisfying accuracy in differentiating conferences with greater impacts from the rest of the crowd. Evaluation results also showed that our heuristics were effective in detecting some extremely low-quality conferences. The same heuristics were also applied to rank conferences, which produced reasonably good results and was able to discover emergent venues of good quality.

We believe this study filled in a gap in the current bibliometrics research by introducing some novel quality measures of publication venues. The findings of this work have a number of implications. They shed light on the patterns of PC members in reputable conferences as well as low-quality ones. As conferences become increasingly prevalent as the major outlet for publication in Computer Science, the impact of such quality indicators will be even more significant. The outcome of this study can be directly applied in existing digital libraries to complement and enhance the existing bibliometrics for ranking and recommending reputable publishing venues.

7.9 A Prototype System: Automatic Conference Quality Rating

In this section, we present an overview of the online prototype conference ranking system, available at:

`http://heavenly.ist.psu.edu:8080/confrating.html`

Figure 7.12 shows a screen shot of the prototype system, in which the Program Committees of the ACM SIGMOD 2007 are shown.

The prototype system takes a conference (and the corresponding list of Program Committee members) as input from the user, and outputs a *quality rating*: whether this conference is a highly reputable conference. The prototype consists of the following components. The *PC Database* contains a list of conferences and the corresponding Program Committee members. For each of these PC members, the *Features Database* crawls the Internet, and collects the information needed to compute the features for the classifier from a number of external data sources,

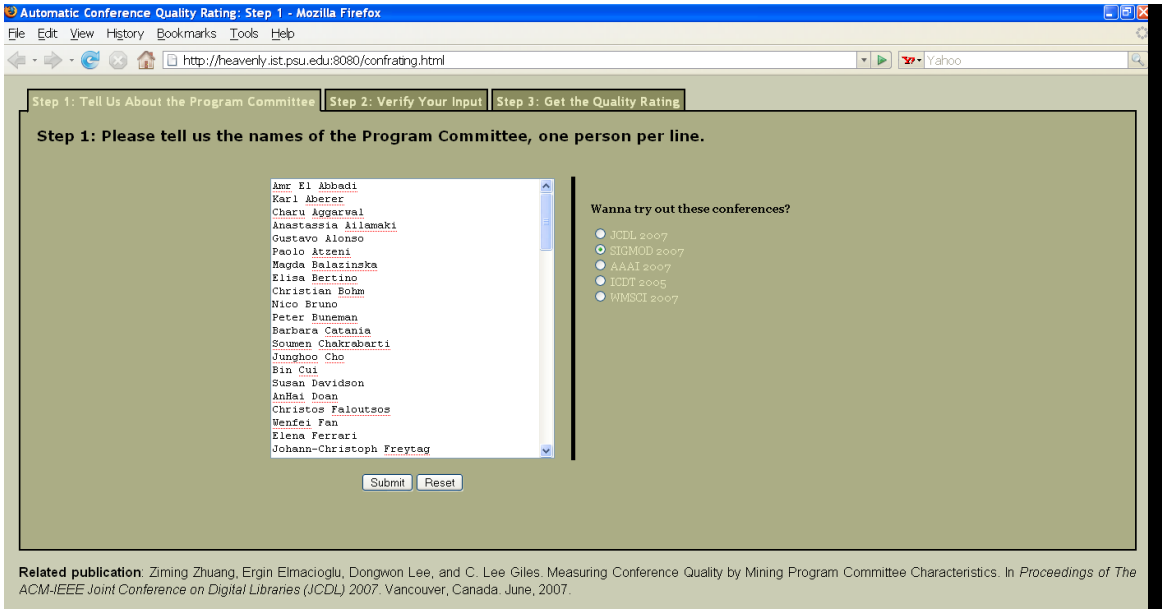

Figure 7.12. A screen shot of the online prototype conference ranking system.

such as the ACM Digital Library[11], the CiteSeer Digital Library[12], and the DBLP Computer Science Bibliography[13]. The *Name Parser* parses the user-supplied list of Program Committee members, cross-checks the names with the *PC Database*: if the names exactly match those of a conference that has already been assigned a reputation rating, the system will output the cached rating immediately. Otherwise, the *Feature Builder* is triggered, and pulls the data from the *Features Database* to generate the feature values required as the input to the *Classifier*. Then the *Classifier* classifies the conference, assigns a reputation rating according the classification label and the confidence, and outputs the rating to the user. Figure 7.13 illustrates the architecture of the prototype system.

[11]http://portal.acm.org
[12]http://citeseerx.ist.psu.edu
[13]http://dblp.uni-trier.de/

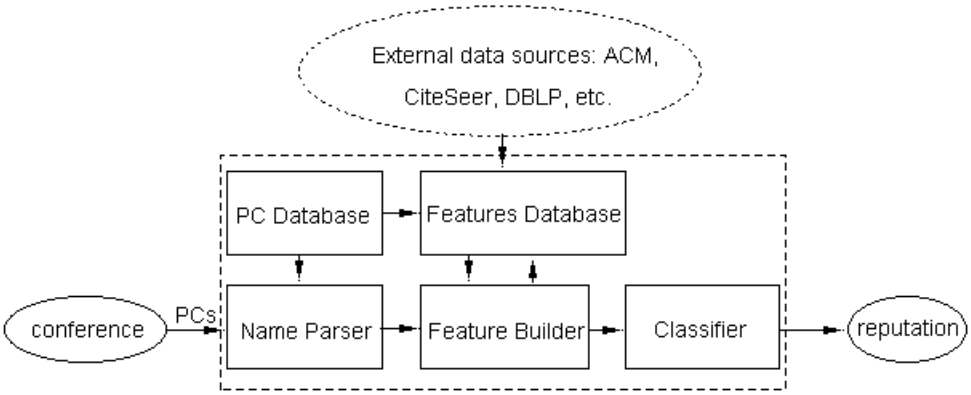

Figure 7.13. Architecture overview of the online prototype conference ranking system.

Chapter 8

Conclusion and Future Work

We studied the use of feedback in relevance ranking and recommendation systems. Specifically, we first presented a flow-based network model to leverage the feedback in a collaborative search and ranking process. Then we focus our lens to investigate the implicit user feedback embedded in their query reformulations, and described a method to utilize this knowledge to improve the relevance ranking for a general purpose Web search engine. Afterwards, we presented two case studies on verticals. In the first case study, we present our findings on using user clicks as their implicit feedback to infer locality preference in geographic information retrieval. In the second case study, we exploit the feedback loop between the publication venues and their organizers (i.e., Program Committees) to derive heuristics for a venue recommendation system that can be used in the context of digital libraries.

We believe there are a number of directions for applications of our work. The first direction is personalization. So far we have investigated the utility of *mass* feedback – query reformulations and clicks – drawn from a large number of users in the search engine logs. However, we can also bias our ranking and recommendation towards more *personal* feedback by weighing an individual user's access patterns, with a back-off strategy such that the more general logs are used only when the individual data is not immediately available.

The second direction is to study the temporal impact on the utility of such feedback. Certain feedback take time to accumulate (e.g., citation records), some tend to be consistent over time (e.g., the user clicks on the homepage of *Disneyland*), while others change as time goes by (e.g., the search queries and clicks on the news

about an incoming hurricane). We can exploit the recency factor of different types of feedback according to their temporal sensitivity, e.g., weighting the feedback by a time-decay function.

The third direction is to expand our study on the feedback loop between the venues and the organizers to include the users of the venue recommendation system. For example, given a topic, who are more likely to search for venues in this topic? What are the typical search queries? If a user is a past collaborator of a Program Committee member, shall we recommend the conference to this user? Eventually, we can recommend certain conferences based on the social distance and similarity in interests between the user and the Program Committee members.

Appendix A

Additional Remarks on Several Ranking Scenarios for FlowRank

In the *FlowRank* model, the relevance ranking of a set of documents D^t is associated with the flow values between the target query q^t and each of the document $d_i^t \in D^t$, denoted by $F(q^t, d_i^t)$.

We discuss in this section the possible values of $F(q^t, d_i^t)$, and limit our scope to a number of representative ranking scenarios.

Lemma 2. *$F(q^t, d_i^t) \in [0, 1]$, if $\forall e \in E, 0 \leq C(e) \leq 1$.*

Proof. According to the definition of network flows, $\forall e \in E$, $F(e)$ is bounded by $C(e)$ such that $0leqF(e) \leq C(e)$. Because $\forall e \in E, C(e) \leq 1, 0 \leq F(e) \leq C(e) \leq 1$. Therefore, $F(q^t, d_i^t) \in [0, 1]$. □

Thus, we discuss the possible values of $F(q^t, d_i^t)$ as follows.

(I). $F(q^t, d_i^t) = 0$ or $F(q^t, d_i^t) = 1$. These are two extreme conditions. $F(q^t, d_i^t) = 0$ represents the scenario in which a document should not be retrieved at all, because it bears no relevance to the target query. On the other hand, $F(q^t, d_i^t) = 1$ represents the scenario in which we have both the maximal relevance between the queries and the document, and the maximal similarity between the collaborating users.

(II). $0 < F(q^t, d_i^t) \leq F(q^t, d_j^t) < 1$**.** Let us first draw a simplification of $C(u_i, u^t) = 1$, which intuitively indicates that all the collaborators are identical. Consider the following two scenarios:

1. $F(q^t, d_i^t) \leq C(q^t, d_i^t) \leq F(q^t, d_j^t) \leq C(q^t, d_j^t)$.
   $C(q^t, d_i^t) = sim(q^t, d_i^t) \leq C(q^t, d_j^t) = sim(q^t, d_j^t)$, which means d_j^t is more relevant to q^t than d_i^t. In this case, we rank d_j^t higher than d_i^t. The ranking problem is reduced to the baseline scenario, in which the query-document similarity is calculated and the documents are ranked based on their relevance to the target query (the query-document similarity).
2. $F(q^t, d_i^t) \leq F(q^t, d_j^t) \leq C(q^t, d_i^t) \leq C(q^t, d_j^t)$ or $F(q^t, d_i^t) \leq F(q^t, d_j^t) \leq C(q^t, d_j^t) \leq C(q^t, d_i^t)$.
   In these two cases, the flows between q^t and D^t are only bounded by $C(e_t) \in E_t, e_t \in E_t$ is on the route of $F(d_n^t, u^t)$. Because of the prior simplification, $C(u_i, u^t)$ has no contribution so it can be excluded for consideration. Because of Lemma 1, we have

$$\forall d_i^t \in D^t, F(q^t, d_i^t) = \sum_{\exists edge(q_i, d_i^t)} C(q_i, d_i^t) + \sum_{\exists edge(u_i, d_i^t)} C(u_i, d_i^t), \quad \text{(A.1)}$$

so that

$$\begin{aligned} F(q^t, d_i^t) - F(q^t, d_i^t) = & \left[\textstyle\sum_{\exists edge(u_j, d_j^t)} C(u_j, d_j^t) - \sum_{\exists edge(u_i, d_i^t)} C(u_i, d_i^t)\right] \\ & + \left[\textstyle\sum_{\exists edge(q_j, d_j^t)} C(q_j, d_j^t) - \sum_{\exists edge(q_i, d_i^t)} C(q_i, d_i^t)\right] \end{aligned} \quad \text{(A.2)}$$

On the other hand, $\sum_{\exists edge(u_j, d_j^t)} C(u_j, d_j^t) - \sum_{\exists edge(u_i, d_i^t)} C(u_i, d_i^t)$ is computed when only the arcs between D^t and U' are considered. Thus,

$$\begin{aligned} & \sum_{\exists edge(u_j, d_j^t)} C(u_j, d_j^t) - \sum_{\exists edge(u_i, d_i^t)} C(u_i, d_i^t) = \\ & sim(q^t, d_j^t) \cdot \sum \left\{ P(d_j^t | u_j) \cdot [1 - P(d_j^t | \overline{u_j})] \right\} \end{aligned}$$

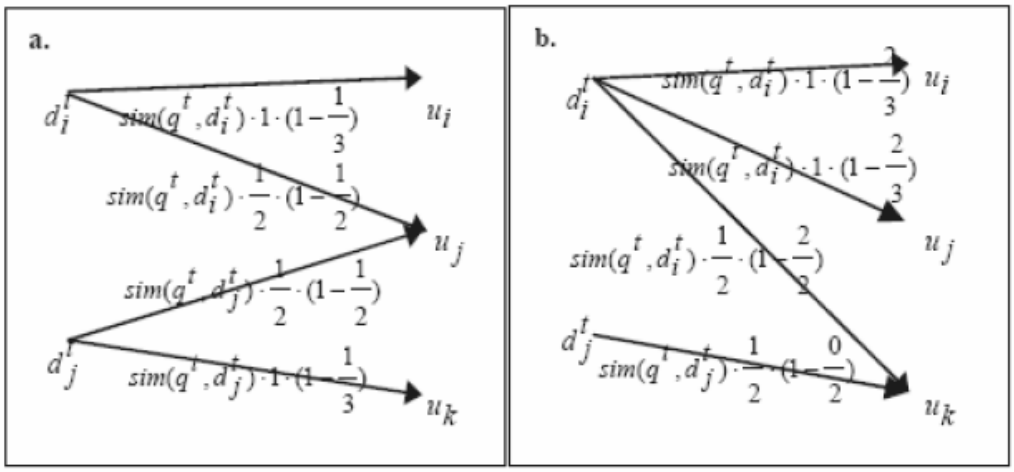

Figure A.1. Different users submitted the target query. Scenario (a): d_i^t and d_j^t have the same probability of receiving user clicks. Scenario (b): d_i^t has a higher probability of receiving user clicks.

$$-sim(q^t, d_i^t) \cdot \sum \left\{ P(d_i^t|u_i) \cdot [1 - P(d_i^t|\overline{u_i})] \right\}. \tag{A.3}$$

Consider the following scenarios. In Figure A.1 (a), d_i^t and d_j^t have the same probability of receiving user clicks, so we have $F(q^t, d_j^t) - F(q^t, d_i^t = (11/12) \cdot [sim(q^t, d_j^t) - sim(q^t, d_i^t)]$ which solely depends on the query-document similarities. On the other hand, in Figure A.1 (b), d_i^t has a higher probability of receiving user clicks, so $F(q^t, d_j^t) - F(q^t, d_i^t) = (1/2) \cdot sim(q^t, d_j^t) - (2/3) \cdot sim(q^t, d_i^t)$; apparently d_i^t is favored in this case.

In what has been discussed so far, if only the first part of Equation (I) is considered, *FlowRank* reduces to a collaborative ranking algorithm that ranks the documents according to their probability of being visited by the users. We call it a collective random surfer model.

Now we discuss the second part of equation (I). According to (3) in Chapter 4.3, we have $C(q_i, d_i^t) = sim(q_i, d_i^t) \cdot sim(q_i, q^t) \cdot P(d_i^t|U_i)$. Thus,

$$\sum_{\exists edge(q_j, d_j^t)} C(q_j, d_j^t) - \sum_{\exists edge(q_i, d_i^t)} C(q_i, d_i^t) =$$

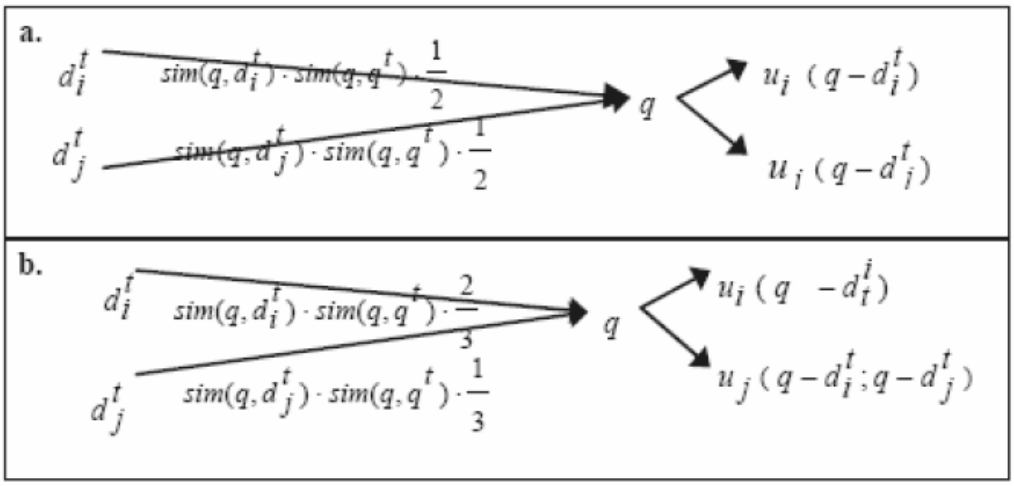

Figure A.2. Different users submitted the same query which was similar to the target query. The notations in the brackets denote the submitted query and the document(s) that a user clicked on. Scenario (a): d_i^t and d_j^t have the same probability of receiving user clicks. Scenario (b): d_i^t has a higher probability of receiving user clicks.

$$\sum_{\exists edge(q_j, d_j^t)} \left[sim(q_j, d_j^t) \cdot sim(q_j, q^t) \cdot P(d_j^t | U_j) \right] - \sum_{\exists edge(q_j, d_i^t)} \left[sim(q_i, d_i^t) \cdot sim(q_i, q^t) \cdot P(d_i^t | U_i) \right]. \quad \text{(A.4)}$$

In Figure A.2 (a) and (b), there are two different users submitting the same query. In (a), d_i^t and d_j^t the same probability of receiving user clicks; both are equally favored by the algorithm. On the other hand, in (b) d_i^t has a higher probability of receiving user clicks, so the algorithm favors d_i^t.

In Figure A.3, both d_i^t and d_j^t have the same probability of receiving user clicks via two different queries issued by two different uses. Note that here q_1 and q_2 are two different queries, both somewhat similar to the target query. The two documents are equally favored. The difference between the arc capacities are decided by the query-document similarities between q_1 / q_2 and the documents, and query-query similarities between q_1 / q_2 and q^t. In Figure A.4, on the other hand, d_j^t has a higher chance of being visited, and is favored by the algorithm.

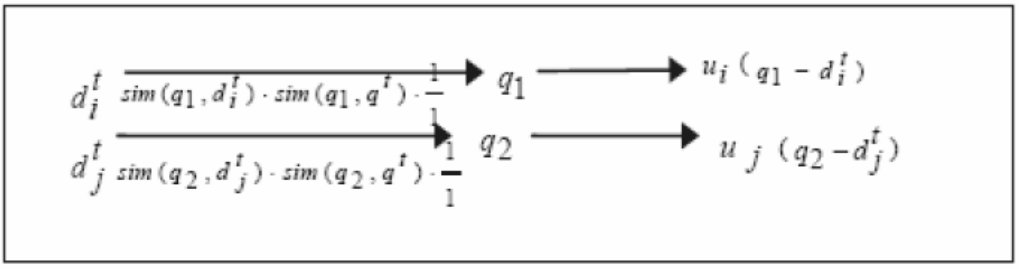

Figure A.3. Different users submitted different queries which were both similar to the target query. The difference between the arc capacities are decided by the query-query similarities and the query-document similarities.

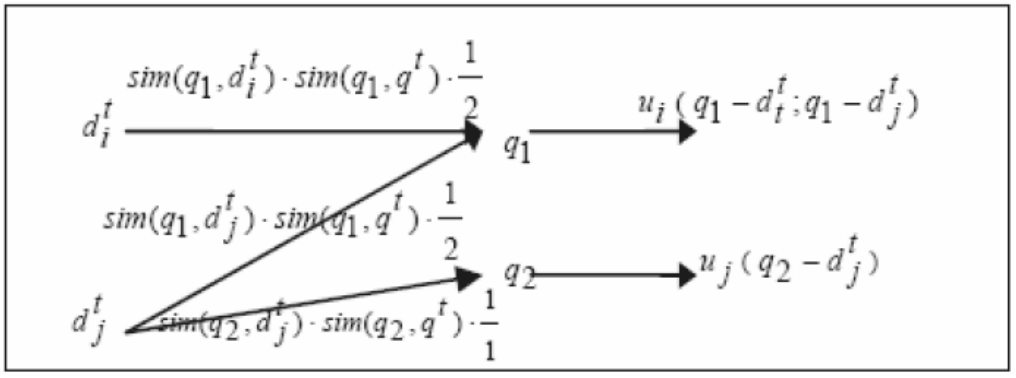

Figure A.4. Different users submitted different queries which were similar to the target query. d_j^t is favored.

Appendix B

The Decision Tree Classifier Model

We present below the complete decision tree used to produce the classification results presented in Chapter 7, Table 7.2 and Table 7.3.

```
coauthoravg <= 12.545455
| pccount <= 34
| | pubavg <= 0: C (100.0)
| | pubavg > 0
| | | coauthoravg <= 4.555556
| | | | pccount <= 27: C (136.0/3.0)
| | | | pccount > 27
| | | | | closeness_avg <= 0.016716: R (3.0)
| | | | | closeness_avg > 0.016716: C (5.0)
| | | coauthoravg > 4.555556
| | | | closeness_avg <= 0.043088
| | | | | closeness_avg <= 0.039842: C (55.0/5.0)
| | | | | closeness_avg > 0.039842
| | | | | | pubavg <= 13
| | | | | | | pccount <= 10: C (13.0)
| | | | | | | pccount > 10
| | | | | | | | betweenness_avg <= 0.000038
```

```
| | | | | | | | | closeness_avg <= 0.040028: R (6.0)
| | | | | | | | | closeness_avg > 0.040028
| | | | | | | | | | pccount <= 27
| | | | | | | | | | | closeness_avg <= 0.042751
| | | | | | | | | | | | pubavg <= 6
| | | | | | | | | | | | | pccount <= 14: R (4.0)
| | | | | | | | | | | | | pccount > 14
| | | | | | | | | | | | | | coauthoravg <= 4.904762: R (2.0)
| | | | | | | | | | | | | | coauthoravg > 4.904762: C (12.0/2.0)
| | | | | | | | | | | | pubavg > 6: C (5.0)
| | | | | | | | | | | closeness_avg > 0.042751: R (9.0/1.0)
| | | | | | | | | | pccount > 27: C (7.0)
| | | | | | | | betweenness_avg > 0.000038: C (4.0)
| | | | | | pubavg > 13: R (5.0)
| | | | closeness_avg > 0.043088
| | | | | pubavg <= 5
| | | | | | coauthoravg <= 4.625: R (4.0)
| | | | | | coauthoravg > 4.625: C (61.0/11.0)
| | | | | pubavg > 5: C (625.0/72.0)
| pccount > 34: C (324.0/5.0)
coauthoravg > 12.545455
| pccount <= 100
| | pccount <= 25
| | | pccount <= 22
| | | | pubavg <= 27
| | | | | pubavg <= 14
| | | | | | pubavg <= 13
| | | | | | | pubavg <= 12
| | | | | | | | pubavg <= 10
| | | | | | | | | coauthoravg <= 13.3
| | | | | | | | | | pccount <= 17: R (2.0)
| | | | | | | | | | pccount > 17: C (4.0)
| | | | | | | | | coauthoravg > 13.3: C (16.0)
```

```
| | | | | | | | pubavg > 10
| | | | | | | | | coauthoravg <= 14.909091
| | | | | | | | | | coauthoravg <= 14.666667
| | | | | | | | | | | coauthoravg <= 13.666667
| | | | | | | | | | | | closeness_avg <= 0.063025: R (4.0)
| | | | | | | | | | | | closeness_avg > 0.063025: C (5.0/1.0)
| | | | | | | | | | | coauthoravg > 13.666667: C (5.0)
| | | | | | | | | | coauthoravg > 14.666667: R (9.0)
| | | | | | | | | coauthoravg > 14.909091: C (11.0/1.0)
| | | | | | | pubavg > 12
| | | | | | | | coauthoravg <= 15.833333: C (35.0)
| | | | | | | | coauthoravg > 15.833333
| | | | | | | | | closeness_avg <= 0.063801: R (4.0)
| | | | | | | | | closeness_avg > 0.063801: C (7.0)
| | | | | | pubavg > 13
| | | | | | | pccount <= 21
| | | | | | | | betweenness_avg <= 0.000073
| | | | | | | | | pccount <= 17
| | | | | | | | | | pccount <= 8
| | | | | | | | | | | coauthoravg <= 15.230769: R (3.0)
| | | | | | | | | | | coauthoravg > 15.230769: C (2.0)
| | | | | | | | | | pccount > 8: C (19.0/3.0)
| | | | | | | | | pccount > 17
| | | | | | | | | | coauthoravg <= 16.806122: R (11.0)
| | | | | | | | | | coauthoravg > 16.806122: C (2.0)
| | | | | | | | betweenness_avg > 0.000073: R (5.0)
| | | | | | | pccount > 21: C (9.0)
| | | | | pubavg > 14
| | | | | | pccount <= 13
| | | | | | | coauthoravg <= 18.294118: C (76.0/1.0)
| | | | | | | coauthoravg > 18.294118
| | | | | | | | coauthoravg <= 20.666667
| | | | | | | | | coauthoravg <= 20.125
```

```
| | | | | | | | | | pubavg <= 18: R (3.0/1.0)
| | | | | | | | | | pubavg > 18
| | | | | | | | | | | pubavg <= 23: C (12.0)
| | | | | | | | | | | pubavg > 23
| | | | | | | | | | | | pccount <= 9: R (2.0)
| | | | | | | | | | | | pccount > 9: C (3.0)
| | | | | | | | | coauthoravg > 20.125: R (8.0/1.0)
| | | | | | | | coauthoravg > 20.666667: C (70.0/7.0)
| | | | | | pccount > 13
| | | | | | | closeness_avg <= 0.069962: C (169.0/22.0)
| | | | | | | closeness_avg > 0.069962
| | | | | | | | closeness_avg <= 0.076687
| | | | | | | | | coauthoravg <= 23.653846
| | | | | | | | | | pubavg <= 22
| | | | | | | | | | | coauthoravg <= 15.333333: R (6.0)
| | | | | | | | | | | coauthoravg > 15.333333
| | | | | | | | | | | | pubavg <= 21
| | | | | | | | | | | | | pccount <= 17
| | | | | | | | | | | | | | coauthoravg <= 18.896104
| | | | | | | | | | | | | | | closeness_avg <= 0.071275: R (4.0)
| | | | | | | | | | | | | | | closeness_avg > 0.071275: C (4.0)
| | | | | | | | | | | | | | coauthoravg > 18.896104: R (4.0)
| | | | | | | | | | | | | pccount > 17: C (7.0)
| | | | | | | | | | | | pubavg > 21: R (3.0)
| | | | | | | | | | pubavg > 22: C (14.0/1.0)
| | | | | | | | | coauthoravg > 23.653846: R (11.0)
| | | | | | | | closeness_avg > 0.076687: C (31.0/2.0)
| | | | pubavg > 27
| | | | | betweenness_avg <= 0.000188
| | | | | | coauthoravg <= 30.043478
| | | | | | | pccount <= 18
| | | | | | | | betweenness_avg <= 0.000133
| | | | | | | | | pubavg <= 33
```

```
| | | | | | | | | | pubavg <= 32
| | | | | | | | | | | coauthoravg <= 20.351852: C (6.0)
| | | | | | | | | | | coauthoravg > 20.351852
| | | | | | | | | | | | coauthoravg <= 22.571429: R (6.0)
| | | | | | | | | | | | coauthoravg > 22.571429
| | | | | | | | | | | | | pccount <= 14: C (12.0/3.0)
| | | | | | | | | | | | | pccount > 14: R (4.0)
| | | | | | | | | | pubavg > 32: R (7.0)
| | | | | | | | | pubavg > 33: C (4.0)
| | | | | | | | betweenness_avg > 0.000133: R (10.0)
| | | | | | | pccount > 18
| | | | | | | | closeness_avg <= 0.070794: C (24.0/2.0)
| | | | | | | | closeness_avg > 0.070794
| | | | | | | | | closeness_avg <= 0.078637: R (12.0/1.0)
| | | | | | | | | closeness_avg > 0.078637: C (3.0)
| | | | | | coauthoravg > 30.043478: C (33.0/3.0)
| | | | | betweenness_avg > 0.000188
| | | | | | closeness_avg <= 0.090293
| | | | | | | closeness_avg <= 0.066591
| | | | | | | | pccount <= 11: C (10.0)
| | | | | | | | pccount > 11: R (5.0)
| | | | | | | closeness_avg > 0.066591: C (24.0)
| | | | | | closeness_avg > 0.090293: R (3.0)
| | | pccount > 22
| | | | closeness_avg <= 0.056423
| | | | | closeness_avg <= 0.055199: C (5.0/1.0)
| | | | | closeness_avg > 0.055199: R (4.0)
| | | | closeness_avg > 0.056423: C (100.0/5.0)
| | pccount > 25
| | | pubavg <= 15
| | | | pubavg <= 12
| | | | | pccount <= 43
| | | | | | pccount <= 31: C (12.0)
```

```
| | | | | | pccount > 31
| | | | | | | pubavg <= 10: C (4.0)
| | | | | | | pubavg > 10
| | | | | | | | coauthoravg <= 13.625
| | | | | | | | | closeness_avg <= 0.061991
| | | | | | | | | | betweenness_avg <= 0.000039: C (2.0)
| | | | | | | | | | betweenness_avg > 0.000039: R (7.0)
| | | | | | | | | closeness_avg > 0.061991: C (7.0)
| | | | | | | | coauthoravg > 13.625: R (6.0)
| | | | | pccount > 43: C (17.0)
| | | | pubavg > 12
| | | | | pccount <= 31
| | | | | | coauthoravg <= 13.470588: C (9.0)
| | | | | | coauthoravg > 13.470588
| | | | | | | coauthoravg <= 14.666667: R (17.0)
| | | | | | | coauthoravg > 14.666667
| | | | | | | | pubavg <= 13: C (6.0/1.0)
| | | | | | | | pubavg > 13
| | | | | | | | | coauthoravg <= 18
| | | | | | | | | | pubavg <= 14
| | | | | | | | | | | coauthoravg <= 16.642857: R (4.0)
| | | | | | | | | | | coauthoravg > 16.642857: C (2.0)
| | | | | | | | | | pubavg > 14
| | | | | | | | | | | coauthoravg <= 16
| | | | | | | | | | | | closeness_avg <= 0.064945: R (3.0/1.0)
| | | | | | | | | | | | closeness_avg > 0.064945: C (2.0)
| | | | | | | | | | | coauthoravg > 16: R (13.0)
| | | | | | | | | coauthoravg > 18: C (3.0)
| | | | | pccount > 31
| | | | | | coauthoravg <= 14.909091
| | | | | | | closeness_avg <= 0.069664
| | | | | | | | coauthoravg <= 14.689655
| | | | | | | | | pubavg <= 14
```

```
| | | | | | | | | | pubavg <= 13: C (4.0/1.0)
| | | | | | | | | | pubavg > 13
| | | | | | | | | | | pccount <= 60: R (26.0/3.0)
| | | | | | | | | | | pccount > 60: C (2.0)
| | | | | | | | | pubavg > 14
| | | | | | | | | | pccount <= 82: C (16.0)
| | | | | | | | | | pccount > 82: R (3.0)
| | | | | | | | coauthoravg > 14.689655: R (10.0)
| | | | | | | closeness_avg > 0.069664: C (8.0)
| | | | | | coauthoravg > 14.909091
| | | | | | | pccount <= 39
| | | | | | | | pubavg <= 13: R (4.0)
| | | | | | | | pubavg > 13
| | | | | | | | | coauthoravg <= 16.92: C (21.0/1.0)
| | | | | | | | | coauthoravg > 16.92
| | | | | | | | | | closeness_avg <= 0.056423: C (4.0)
| | | | | | | | | | closeness_avg > 0.056423: R (6.0)
| | | | | | | pccount > 39: C (23.0)
| | | pubavg > 15
| | | | pubavg <= 17
| | | | | pccount <= 36: C (68.0/2.0)
| | | | | pccount > 36
| | | | | | coauthoravg <= 17.96875
| | | | | | | pubavg <= 16
| | | | | | | | betweenness_avg <= 0.000059: C (7.0)
| | | | | | | | betweenness_avg > 0.000059
| | | | | | | | | coauthoravg <= 14.727273: R (2.0)
| | | | | | | | | coauthoravg > 14.727273
| | | | | | | | | | pccount <= 42: R (5.0/2.0)
| | | | | | | | | | pccount > 42: C (2.0)
| | | | | | | pubavg > 16
| | | | | | | | betweenness_avg <= 0.000071
| | | | | | | | | closeness_avg <= 0.06218: R (5.0/1.0)
```

```
| | | | | | | | | closeness_avg > 0.06218: C (17.0)
| | | | | | | | betweenness_avg > 0.000071: C (21.0)
| | | | | | coauthoravg > 17.96875: R (7.0/1.0)
| | | | pubavg > 17
| | | | | pubavg <= 19
| | | | | | pccount <= 38
| | | | | | | closeness_avg <= 0.069296: R (25.0/1.0)
| | | | | | | closeness_avg > 0.069296
| | | | | | | | betweenness_avg <= 0.000063: C (6.0)
| | | | | | | | betweenness_avg > 0.000063: R (6.0)
| | | | | | pccount > 38
| | | | | | | coauthoravg <= 18.6: C (30.0/1.0)
| | | | | | | coauthoravg > 18.6
| | | | | | | | pccount <= 52: C (4.0)
| | | | | | | | pccount > 52: R (9.0/1.0)
| | | | | pubavg > 19
| | | | | | pccount <= 71
| | | | | | | coauthoravg <= 17.939394: C (26.0)
| | | | | | | coauthoravg > 17.939394
| | | | | | | | coauthoravg <= 18.461538
| | | | | | | | | closeness_avg <= 0.060216: R (7.0)
| | | | | | | | | closeness_avg > 0.060216: C (3.0)
| | | | | | | | coauthoravg > 18.461538
| | | | | | | | | closeness_avg <= 0.056741
| | | | | | | | | | pccount <= 40
| | | | | | | | | | | pccount <= 28: R (2.0)
| | | | | | | | | | | pccount > 28: C (2.0)
| | | | | | | | | | pccount > 40: R (5.0)
| | | | | | | | | closeness_avg > 0.056741
| | | | | | | | | | pubavg <= 26
| | | | | | | | | | | pccount <= 32: C (44.0)
| | | | | | | | | | | pccount > 32
| | | | | | | | | | | | betweenness_avg <= 0.000154
```

```
| | | | | | | | | | | | pccount <= 49
| | | | | | | | | | | | | coauthoravg <= 20.863636: C (18.0/1.0)
| | | | | | | | | | | | | coauthoravg > 20.863636
| | | | | | | | | | | | | | coauthoravg <= 21.484848: R (8.0)
| | | | | | | | | | | | | | coauthoravg > 21.484848
| | | | | | | | | | | | | | | closeness_avg <= 0.066434: R (8.0/1.0)
| | | | | | | | | | | | | | | closeness_avg > 0.066434: C (26.0)
| | | | | | | | | | | | pccount > 49: C (21.0)
| | | | | | | | | | | betweenness_avg > 0.000154: C (10.0)
| | | | | | | | | | pubavg > 26
| | | | | | | | | | closeness_avg <= 0.068604: C (10.0)
| | | | | | | | | | closeness_avg > 0.068604
| | | | | | | | | | | pubavg <= 30
| | | | | | | | | | | | pccount <= 29: R (8.0)
| | | | | | | | | | | | pccount > 29
| | | | | | | | | | | | | coauthoravg <= 27.68: C (5.0/1.0)
| | | | | | | | | | | | | coauthoravg > 27.68: R (4.0/1.0)
| | | | | | | | | | | pubavg > 30: C (5.0/1.0)
| | | | | | pccount > 71
| | | | | | | closeness_avg <= 0.0687: R (10.0)
| | | | | | | closeness_avg > 0.0687: C (2.0)
| pccount > 100
| | closeness_avg <= 0.066859
| | | closeness_avg <= 0.061613
| | | | betweenness_avg <= 0.000048: R (8.0)
| | | | betweenness_avg > 0.000048
| | | | | pubavg <= 19: C (5.0/1.0)
| | | | | pubavg > 19: R (2.0)
| | | closeness_avg > 0.061613: R (49.0/1.0)
| | closeness_avg > 0.066859: C (4.0/1.0)
```

Bibliography

[1] Pew internet report. *Retrieved at http://www.pewinternet.org/reports.asp*, 2007.

[2] The size of the world wide web. *Retrieved at http://www.boutell.com/newfaq/misc/sizeofweb.html*, 2007.

[3] Official google blog: We know the web was big. *Retrieved at http://googleblog.blogspot.com/2008/07/we-knew-web-was-big.html*, 2008.

[4] S. Ahern, M. Naaman, R. Nair, and J. Yang. World explorer: Visualizing aggregate data from unstructured text in geo-referenced collections. In *JCDL '07: Proceedings of the 2007 conference on Digital libraries*, pages 1–10. ACM Press, 2007.

[5] E. Amitay, N. Har'El, R. Sivan, and A. Soffer. Web-a-where: geotagging web content. In *Proc. of the 27th ACM SIGIR conference on Research and development in information retrieval*, pages 273–280, 2004.

[6] L. Backstrom, J. Kleinberg, R. Kumar, and J. Novak. Spatial variation in search engine queries. In *Proc. of the 17th Intl. Conference on World Wide Web*, pages 357–366, 2008.

[7] A. Barabasi, H. Jeong, Z. Neda, E. Ravasz, A. Schubert, and T. Vicsek. Evolution of the social network of scientific collaborations. *Physica A*, 311(3-4):590–614, 2002.

[8] D. Beeferman and A. Berger. Agglomerative clustering of a search engine query log. In *In Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 407–416.

[9] T. Berners-Lee, T. Bray, D. Connolly, P. Cotton, R. Fielding, M. Jeckle, C. Lilley, N. Mendelsohn, D. Orchard, N. Walsh, and S. Williams. Architecture of the world wide web, volume one. *The World Wide Web Consortium*, December 2004.

[10] P. A. Bernstein, D. DeWitt, A. Heuer, Z. Ives, C. S. Jensen, H. Meyer, M. T. Ozsu, R. T. Snodgrass, K.-Y. Whang, and J. Widom. Database publication practices. In *VLDB '05: Proceedings of the 31st international conference on Very large data bases*, pages 1241–1245. VLDB Endowment, 2005.

[11] K. Bharat, A. Broder, M. Henzinger, P. Kumar, and S. Venkatasubramanian. The connectivity server: Fast access to linkage information on the web. In *In Proceedings of the 7th International World Wide Web Conference*, pages 469–477, 1998.

[12] K. Bharat and G. Mihaila. When experts agree: Using non-affiliated experts to rank popular topics. In *In Proceedings of the 10th International World Wide Web Conference*, pages 597–602, 2001.

[13] P. Boldi, F. Bonchi, C. Castillo, D. Donato, A. Gionis, and S. Vigna. The query-flow graph: model and applications. In *In Proceedings of the ACM Conference on Information and Knowledge Management (CIKM)*, 2008.

[14] J. Bollen, H. V. de Sompel, J. A. Smith, and R. Luce. Toward alternative metrics of journal impact: A comparison of download and citation data. *Information Processing & Management*, 41(6):1419–1440, December 2005.

[15] J. Bollen, M. A. Rodriguez, and H. V. de Sompel. *Retrieved at http://www.arxiv.org/abs/cs.GL/0601030*.

[16] F. Bonchi, C. Castillo, D. Donato, and A. Gionis. Topical query decomposition. In *In Proceedings of the ACM KDD Conference*, 2008.

[17] S. Brin, R. Motwani, L. Page, and Winograd. What can you do with a web in your pocket. *IEEE Data Engineering Bulletin*, (21):37–47, 1998.

[18] S. Brin and L. Page. The anatomy of a large scale hypertextual web search engine. In *In Proceedings of the 7th International World Wide Web Conference*, pages 107–117, 1998.

[19] A. Broder, R. Kumar, F. Maghoull, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. Graph structure in the web. In *In Proceedings of the 9th International World Wide Web Conference*, pages 309–320, 2000.

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

[21] G. Cai. Geovsm: An integrated retrieval model for geographic information. In *Geographic Information Science: Second International Conference, GI-Science 2002*, pages 70–85. Springer, 2002.

[22] J. Carriere and R. Kazman. Webquery: Searching and visualizing the web through connectivity. In *In Proceedings of the 6th International World Wide Web Conference*, 1997.

[23] S. Chakrabati, B. Dom, D. Gibson, J. Kleinberg, P. Raghavan, and S. Rajagonpalan. Automatic resource list compilation by analyzing hyperlink structure and associated text. In *In Proceedings of the 7th International World Wide Web Conference*, 1998.

[24] G. Chartrand. Cut-vertices and bridges. *Introductory Graph Theory*, pages 45–49, 1985.

[25] K. Cheung and L. Tian. Learning user similarity and rating style for collaborative recommendation. *Information Retrieval*, 7(3-4):395–410, 2004.

[26] B. Chidlovskii, N. S. Glance, and M. A. Grasso. Collaborative re-ranking of search results. In *In The National Conference on Artificial Intelligence 2000 Workshop on AI for Web Search*, pages 18–23, 2000.

[27] S. Chien and N. Immorlica. Semantic similarity between search engine queries using temporal correlation. In *In Proceedings of the 14th International Conference on World Wide Web*, pages 2–1.

[28] K. P. Chitrapura and S. R. Kashyap. Node ranking in labeled directed graphs. In *In Proceedings of ACM Conference on Information and Knowledge Management*, pages 597–606, 2004.

[29] M. Craven, S. Slattery, and K. Nigam. First-order learning for web mining. In *In Proceedings of the 10th European Conference on Machine Learning (ECML)*, pages 250–255, 1998.

[30] S. Cucerzan and E. Brill. Spelling correction as an iterative process that exploits the collective knowledge of web users. In *In Proceedings of the Empirical Methods in Natural Language Processing Conference (EMNLP 2004)*, pages 293–300, 2004.

[31] H. Cui, J. Wen, J. Nie, and W. Ma. Probabilistic query expansion using query logs. In *In Proceedings of the 11th World Wide Web Conference*, pages 325–332, 2002.

[32] H. Daume and E. Brill. Web search intent induction via automatic query reformulation. In *In Human Language Technology Conference / North American Chapter of the Association for Computational Linguistics*, 2004.

[33] B. Davison, M. Najork, and T. Converse. Sigir workshop report: Adversarial information retrieval on the web (airweb 2006). In *ACM SIGIR Forum*, volume 40, pages 0–3, 2006.

[34] S. Deerwester, S. T. Dumais, G. W. Furnas, T. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of American Society of Information Sciences*, 41(6):391–407, 1990.

[35] T. M. Delboni, K. A. V. Borges, and A. H. F. Laender. Geographic web search based on positioning expressions. In *GIR '05: Proceedings of the 2005 workshop on Geographic information retrieval*, pages 61–64, 2005.

[36] A. Dempster, N. Laird, and D. Rubin. Likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[37] J. Ding, L. Gravano, and N. Shivakumar. Computing geographical scopes of web resources. In *VLDB '00: Proceedings of the 26th International Conference on Very Large Data Bases*, pages 545–556, 2000.

[38] S. Dumais, E. Cutrell, and H. Chen. Optimizing search by showing results in context. In *In Proceedings of SIGCHI Conference*, 2001.

[39] J. Edmonds and R. M. Kapr. Theoretical improvements in the algorithmic efficiency for network flow problems. *Journal of the ACM*, 19:248–264, 1972.

[40] E. Elmacioglu and D. Lee. On six degrees of separation in dblp-db and more. *SIGMOD Record*, 34(2):33–40, 2005.

[41] G. W. Flake, S. Lawrence, and C. L. Giles. Efficient identification of web communities. In *In Proceedings of the Sixth International Conference on Knowledge Discovery and Data Mining*, pages 150–160, 2000.

[42] G. W. Flake, K. Tsioutsiouliklis, and L. Zhukov. Methods for mining web communities: Bibliometric, spectral, and flow. In *Web Dynamics*, 2003.

[43] L. R. J. Ford and D. R. Fulkerson. Maximal flow through a network. *Canadian Journal of Mathematics*, 8:399–404, 1956.

[44] L. C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 40(1):35–41, 1977.

[45] Y. Freund, R. Iyer, R. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, 4:933–969, 2003.

[46] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting,. *Journal of Computer and System Sciences*, 55(1):119–139, August 1997.

[47] J. Freyne, B. Smyth, M. Coyle, E. Balfe, and P. Briggs. Further experiments on collaborative ranking in community-based web search. 21(3-4):229–252, 2004.

[48] X. Fu, J. Budzik, and K. Hammond. Mining navigation history for recommendation. In *In Proceedings of the 5th International Conference on Intelligent User Interfaces*, pages 106–112, 2000.

[49] E. GARFIELD. Citation indexes for science; a new dimension in documentation through association of ideas. *Science*, 122(3159):108–111, July 1955.

[50] A. V. Goldberg and R. E. Tarjan. A new approach to the maximum-flow problem. *Journal of the ACM*, 35(4):921–940, 1998.

[51] L. Gravano, V. Hatzivassiloglou, and R. Lichtenstein. Categorizing web queries according to geographical locality. In *Proc. of the twelfth international conference on Information and knowledge management*, pages 325–333, 2003.

[52] A. Gulli and A. Signorini. The indexable web is more than 11.5 billion pages. In *WWW '05: Special interest tracks and posters of the 14th international conference on World Wide Web*, pages 902–903, 2005.

[53] T. Haveliwala. Topic-sensitive pagerank. In *In Proceedings of the 11th International World Wide Web Conference*, pages 517–526, 2002.

[54] D. Hiemstra. Using language models for information retrieval. Technical report, 2001.

[55] J. E. Hirsch. An index to quantify an individual's scientific research output. *Proceedings of the National Academy of Sciences*, 102(46):16569–16572, November 2005.

[56] Y. Hong, B.-W. On, and D. Lee. System support for name authority control problem in digital libraries: Opendblp approach. In *Research and Advanced Technology for Digital Libraries, 8th European Conference*, volume 3232 of *Lecture Notes in Computer Science*. Springer, September 2004.

[57] J. Jansen and A. Spink. An analysis of web documents retrieved and viewed. In *In Proceedings of the 4th International Conference on Internet Computing*, 2003.

[58] K. Jarvelin and J. Kekalainen. Ir evaluation methods for retrieving highly relevant documents. In *In Proceedings of the 23rd ACM SIGIR Conference*, pages 41–48, 2000.

[59] K. Jarvelin and J. Kekalainen. Ir evaluation methods for retrieving highly relevant documents. In *Proc. of the 23rd Annual ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 41–48, 2000.

[60] K. Jarvelin and J. Kekalainen. Cumulated gain-based evaluation of ir techniques. 20(4):422–446, 2002.

[61] K. Jarvelin and J. Kekalainen. Cumulated gain-based evaluation of ir techniques. In *ACM Transactions on Information Systems (TOIS)*, pages 422–446, 2002.

[62] T. Joachims. Optimizing search engines using clickthrough data. In *In Proceedings of ACM Conference on Knowledge Discovery and Data Mining*, pages 133–142, 2002.

[63] T. Joachims. Unbiased evaluation of retrieval quality using clickthrough data. In *SIGIR Workshop on Mathematical/Formal Methods in Information Retrieval*, 2002.

[64] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. In *In Proceedings of Annual ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 154–161, 2005.

[65] T. Joachims, L. Granka, B. Pan, H. Hembrooke, F. Radlinski, and G. Gay. Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search. *ACM Transactions on Information Systems (TOIS)*, 25(2), April 2007.

[66] R. Jones, W. V. Zhang, B. Rey, P. Jhala, and E. Stipp. Geographic intention and modification in web search. *International Journal of Geographical Information Science*, 22(3):229–246, 2008.

[67] I. Kang and G. Kim. Query type classification for web document retrieval. In *In Proceedings of the 26th ACM SIGIR Conference*, pages 64–71, 2003.

[68] P. Katerattanakul, B. Han, and S. Hong. Objective quality ranking of computing journals. *Communications of the ACM*, 46(10):111–114, 2003.

[69] L. Kennedy, M. Naaman, S. Ahern, R. Nair, and T. Rattenbury. How flickr helps us make sense of the world: context and content in community-contributed media collections. In *MULTIMEDIA '07: Proceedings of the 15th international conference on Multimedia*, pages 631–640, 2007.

[70] L. S. Kennedy and M. Naaman. Generating diverse and representative image search results for landmarks. In *Proc. of the 17th Intl. Conference on World Wide Web*, pages 297–306, 2008.

[71] J. Kleinberg. Authoritative sources in a hyperlinked environment. In *In Proceedings of the 9th ACM SIAM Symposium on Discrete Algorithms*, pages 668–677, 1998.

[72] J. Koenemann and N. J. Belkin. A case for interaction: A study of interactive information retrieval behavior and effectiveness. In *In Proceedings of the Conference on Human Factors in Computing Systems*, pages 205–212, 1996.

[73] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1137–1143. Morgan Kaufmann, 1995.

[74] R. Kraft and J. Zien. Mining anchor text for query refinement. In *In Proceedings of the World Wide Web Conference*, pages 666–674, 2004.

[75] K. J. Lanfear. A spatial overlay ranking method for a geospatial search of text objects. In *U.S. Geological Survey Open-File Report 2006-1279*, 2006.

[76] B. Larsen and P. Ingwersen. Using citations for ranking in digital libraries. In *ACM/IEEE Joint Conference on Digital Libraries, JCDL 2006*. ACM, 2006.

[77] R. R. Larson and P. Frontiera. *Spatial Ranking Methods for Geographic Information Retrieval (GIR) in Digital Libraries*, volume 3232. January 2004.

[78] T. Lau and E. Horvitz. Patterns of search: Analyzing and modeling web query refinement. In *In Proceedings of the Seventh International Conference on User Modeling*, pages 119–128, 1999.

[79] S. Lawrence and C. L. Giles. Searching the world wide web. *Science Magazine*, 280:98–100, 1998.

[80] R. Lee, H. Shiina, H. Takakura, Y. J. Kwon, and Y. Kambayashi. Optimization of geographic area to a web page for two-dimensional range query processing. pages 9–17, 2003.

[81] R. Lee, H. Shiina, T. Tezuka, Y. Yokota, H. Takakura, Y. J. Kwon, and Y. Kambayashi. Map-based range query processing for geographic web search systems. pages 274–283. 2005.

[82] U. Lee, Z. Liu, and J. Cho. Automatic identification of user goals in web search. In *In Proceedings of the 14th World Wide Web Conference*, pages 391–410, 2005.

[83] R. Lempel and S. Moran. The stochastic approach for link-structure analysis (salsa) and the tkc effect. *Computer Networks*, 33(1-6):387–401, 2000.

[84] J. Luxenburger and G. Weikum. Query-log based authority analysis for web information search. In *The 5th International Conference on Web Information Systems Engineering (WISE)*, volume 3306 of *Lecture Notes in Computer Science*, pages 90–101. Springer, 2004.

[85] G. S. Mann, D. M. Mimno, and A. McCallum. Bibliometric impact measures leveraging topic analysis. In *ACM/IEEE Joint Conference on Digital Libraries, JCDL 2006*. ACM, 2006.

[86] I. Menache, S. Mannor, and N. Shimkin. Q-cut - dynamic discovery of sub-goals in reinforcement learning. In *In Proceedings of the 13th European Conference on Machine Learning (ECML)*, pages 295–306, 2002.

[87] M. Mitra, A. Singhal, and C. Buckley. Improving automatic query expansion. In *In Proceedings of the 21st International ACM SIGIR Conference*, pages 206–214, 1998.

[88] U. Nambiar and S. Kambhampati. Providing ranked relevant results for web database queries. In *In Proceedings of the World Wide Web Conference*, pages 314–315, 2004.

[89] S. Nerur, R. Sikora, G. Mangalaraj, and V. Balijepally. Assessing the relative influence of journals in a citation network. *Communications of the ACM*, 48(11):71–74, 2005.

[90] M. E. J. Newman. Who is the best connected scientist? a study of scientific coauthorship networks. *Physical Review E*, 64:016132, 2001.

[91] A. Ntoulas, M. Najork, M. Manasse, and D. Fetterly. Detecting spam web pages through content analysis. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, pages 83–92, New York, NY, USA, 2006. ACM.

[92] S. Olsen. Does search engine's power threaten web's independence? *Retrieved at http://news.com.com/2009-1023-963618.html*, 2002.

[93] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Brining order to the web. In *Technical Report, Stanford University Database Group*, 1998.

[94] Y. Qiu and H. Frei. Concept-based query expansion. In *In Proceedings of the 16th International ACM SIGIR Conference*, pages 160–169, 1993.

[95] J. R. Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[96] R. J. Quinlan. Bagging, boosting, and c4.5. In *Proceedings of the 13th National Conference on Artificial Intelligence Conference (AAAI)*, volume 1, pages 725–730, 1996.

[97] F. Radlinski and T. Joachims. Query chains: Learning to rank from implicit feedback. In *ACM SIGKDD International Conference On Knowledge Discovery and Data Mining (KDD)*, pages 239–248, 2005.

[98] F. Radlinski, M. Kurup, and T. Joachims. How does clickthrough data reflect retrieval quality? In *ACM Conference on Information and Knowledge Management (CIKM)*, 2008.

[99] E. Rahm and A. Thor. Citation analysis of database publications. *SIGMOD Record*, 34(4):48–53, 2005.

[100] K. M. Risvik, T. Mikolajewski, and P. Boros. Query segmentation for web search. In *In Proceedings of the 11th International World Wide Web Conference*, 2003.

[101] S. Robertson and K. Sparck-Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science*, (27):129–146, 1976.

[102] S. Robertson, H. Zaragoza, and M. Taylor. Simple bm25 extension to multiple weighted fields. In *CIKM '04: Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 42–49, New York, NY, USA, 2004. ACM.

[103] S. E. Robertson and D. Hiemstra. Language models and probability of relevance. In *In Proceedings of the first Workshop on Language Modeling and Information Retrieval*, 2001.

[104] S. E. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 232–241, New York, NY, USA, 1994. Springer-Verlag New York, Inc.

[105] S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, and M. Gatford. Okapi at trec-3. In *Overview of the Third Text REtrieval Conference (TREC-3)*, pages 109–126, 1995.

[106] D. Rose and D. Levinson. Understanding user goals in web search. In *In Proceedings of the World Wide Web Conference*, pages 13–19, 2004.

[107] S. Saha, S. Saint, and D. A. Christakis. Impact factor: a valid measure of journal quality? *Journal of the Medical Library Association*, 91(1):42–46, January 2003.

[108] G. Salton. Developments in automatic text retrieval. 253:974–979, 1991.

[109] G. Salton and C. Buckley. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 41:288–297, 1990.

[110] G. Salton and M. J. McGill. Introduction to modern information retrieval. 1983.

[111] M. Sanderson and Y. Han. Search words and geography. In *Proc. of the 4th ACM workshop on Geographical information retrieval*, pages 13–14, 2007.

[112] M. Sanderson and J. Kohler. Analyzing geographic queries, 2004.

[113] D. Santos and M. S. Chaves. The place of place in geographical ir. In *Proc. of the 3rd Workshop on Geographic Information Retrieval (SIGIR'06)*, pages 5–8, 2006.

[114] J. Savoy and D. Vrajitoru. Evaluation of learning schemes used in information retrieval. *Technical Report CR-I-95-02, Faculty of Sciences, University of Neuchatel*, 1996.

[115] S. Schockaert and M. D. Cock. Neighborhood restrictions in geographic ir. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 167–174, 2007.

[116] A. Shashua and A. Levin. Ranking with large margin principle: Two approaches. In *In Proceedings of the Neural Information and Processing Systems (NIPS) Conference*, 2002.

[117] X. Shen, B. Tan, and C. Zhai. Implicit user modeling for personalized search. In *In Proceedings of the 14th ACM International Conference on Information and Knowledge Management (CIKM '05)*, pages 824–831, 2005.

[118] X. Shen and C. Zhai. Exploiting query history for document ranking in interactive information retrieval. In *In Proceedings of the 26th ACM SIGIR Conference*, pages 377–378, 2003.

[119] C. Sheng, W. Hsu, and M.-L. Lee. Discovering geographical-specific interests from web click data. In *Proceedings of the First International Workshop on Location and the Web (LocWeb)*, pages 41–48, April 2008.

[120] B. Smyth, E. Balfe, O. Boydell, K. Bradley, P. Briggs, M. Coyle, and J. Freyne. A live-user evaluation of collaborative web search. In *In Proc. of the 19th International Joint Conference on Artificial Intelligence (IJCAI'05)*, 2005.

[121] K. Sugiyama, K. Hatano, and M. Yoshikawa. Adaptive web search based on user profile constructed without any effort from users. In *In Proceedings of the 13th International World Wide Web Conference*, pages 675–684, 2004.

[122] J. Teevan, S. Dumais, and E. Horvitz. Personalizing search via automated analysis of interests and activities. In *In Proceedings of the 28th ACM SIGIR Conference*, pages 391–410, 2005.

[123] P. Vakkari. Subject knowledge, source of terms and term selection in query expansion. In *In Proceedings of the 24th European Conference in Information Retrieval*, 2002.

[124] C. Wang, X. Xie, L. Wang, Y. Lu, and W.-Y. Ma. Detecting geographic locations from web resources. In *GIR '05: Proceedings of the 2005 workshop on Geographic information retrieval*, pages 17–24, New York, NY, USA, 2005. ACM Press.

[125] C. Wang, X. Xie, L. Wang, Y. Lu, and W.-Y. Ma. Web resource geographic location classification and detection. In *WWW '05: Special interest tracks and posters of the 14th international conference on World Wide Web*, pages 1138–1139, New York, NY, USA, 2005. ACM Press.

[126] L. Wang, C. Wang, X. Xie, J. Forman, Y. Lu, W.-Y. Ma, and Y. Li. Detecting dominant locations from search queries. In *Proc. of the 28th ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 424–431, 2005.

[127] S. Wasserman and K. Faust. *Social Networks Analysis: Methods and Applications*. Cambridge University Press, United Kingdom, 1994.

[128] J. Wen, J. Nie, and H. Zhang. Clustering user queries of a search engine. In *In Proceedings of the 10th International World Wide Web Conference*, pages 162–168, 2001.

[129] G. Xue, H. Zeng, Z. Chen, Y. Yu, W. Ma, W. Xi, and W. Fan. Optimizing web search using web click-through data. In *In Proceedings of the 13th ACM International Conference on Information and Knowledge Management (CIKM '04)*, pages 118–126, 2004.

[130] O. R. Zaiane and A. Strilets. Finding similar queries to satisfy searches based on query traces. In *In Proceedings of the International Workshop on Efficient Web-Based Information Systems*, pages 207–216, 2002.

[131] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems*, 22(2):179–214, 2004.

[132] Q. Zhang, X. Xie, L. Wang, L. Yue, and W.-Y. Ma. Detecting geographical serving area of web resources. In *Proc. of the 3th ACM workshop on Geographical information retrieval*, 2006.

[133] V. W. Zhang, B. Rey, E. Stipp, and R. Jones. Geomodification in query rewriting. In *Proc. of the 3th ACM SIGIR Workshop on GIR*, 2006.

[134] S. Zhu, X. Ji, W. Xu, and Y. Gong. Multi-labelled classification using maximum entropy method. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 274–281, New York, NY, USA, 2005. ACM.

[135] Z. Zhuang. ihits: Extending hits for personal interests profiling. In *In Proceedings of 19th International Conference on Advanced Information Networking and Applications*, 2005.

[136] Z. Zhuang, C. Brunk, and C. L. Giles. Modeling and visualizing geo-sensitive queries based on user clicks. In S. Boll, C. Jones, E. Kansa, P. Kishor, M. Naaman, R. Purves, A. Scharl, and E. Wilde, editors, *In Proceedings of the First ACM International Workshop on Location and the Web at WWW 2008*, volume 300. ACM, 2008.

[137] Z. Zhuang, C. Brunk, P. Mitra, and C. L. Giles. Towards click-based models of geographic interests in web search. In *In Proceedings of the ACM/IEEE/WIC International Conference on Web Intelligence (WI)*, 2008.

[138] Z. Zhuang and S. Cucerzan. Re-ranking search results using query logs. In *In Proceedings of the 15th ACM international conference on Information and knowledge management*, New York, NY, USA, 2006. ACM.

[139] Z. Zhuang and S. Cucerzan. Exploiting semantic query context to improve search ranking. In *In Proceedings of the 2nd IEEE International Conference on Semantic Computing, ICSC 2008*. IEEE, 2008.

[140] Z. Zhuang, S. Cucerzan, and C. L. Giles. Network flow for collaborative ranking. In J. Fürnkranz, T. Scheffer, and M. Spiliopoulou, editors, *Knowledge Discovery in Databases: PKDD 2006, 10th European Conference on Principles and Practice of Knowledge Discovery in Databases*, volume 4213 of *Lecture Notes in Computer Science*, pages 434–445. Springer, 2006.

[141] Z. Zhuang, E. Elmacioglu, D. Lee, and C. L. Giles. Measuring conference quality by mining program committee characteristics. In E. M. Rasmussen, R. R. Larson, E. Toms, and S. Sugimoto, editors, *ACM/IEEE Joint Conference on Digital Libraries, JCDL 2007*, pages 225–234. ACM, 2007.

Vita

Ziming Zhuang

Ziming Zhuang was born in Guangzhou, China. He received his B.S. degree in Computer Science and Engineering from Fudan University in 2003, and his M.S. degree in Information Sciences and Technology from The Pennsylvania State University in 2008. His current research focuses on data mining and machine learning applications in search relevance ranking features and algorithms, personalization and recommendation, and social network analysis.