IMPROVING WEB SEARCH PERSONALIZATION USING LUHN-INSPIRED VECTOR RE-WEIGHTING

HANZE LIU

Improving Web Search Personalization Using Luhn-Inspired Vector Re-Weighting

by

© Hanze Liu

A thesis submitted to the

School of Graduate Studies

in partial fulfilment of the

requirements for the degree of

Master of Science

Department of Computer Science

Memorial University of Newfoundland

January 2012

St. John's Newfoundland

Abstract

Web search personalization has been studied as a way to tailor Web search results to individual users based on their interests and preferences. Commonly, document and personalization profile features are stored in vector space models using measures such as term frequency (TF) and term frequency-inverse document frequency (TF*IDF). Inspired by Luhn's model of term importance, a novel approach is proposed in this thesis to identify and re-weight significant terms in the vector-based personalization models. Evaluations with a set of ambiguous queries illustrate that the order of the search results using this approach is superior to the TF approach and comparable to the TF*IDF approach. However, it is based only on the information stored in the personalization profiles, rather than requiring access to the distribution of each term across the document collection. As such, it can be applied more broadly when only limited information regarding the collection being searched is available.

Acknowledgements

First and foremost, I would like to give thanks to my supervisor, Dr. Orland Hoeber, for the great help and guidance he offered to me during my Master's program. The research presented in this thesis would not have been possible without his support and encouragement. I also want to thank the Department of Computer Science and Memorial University, who provided generous resources for my study and research. This research is funded by my supervisor's NSERC Discovery Grant and I have received scholarship from the School of Graduate Studies of Memorial University.

Second, I would like to thank my wife, Xia Li, who took the onerous responsibility of taking care of the housework and the new born baby Angela Anqi Liu while I was busy with my research. Her love and encouragement always bring me great comfort when I encountered difficulties in the research and the thesis writing.

Last but not least, my love and thanks go to my parents, Zhicun Liu and Qingxin Xu, who raised me with unconditional love and never-ending patience. I want to give special thanks to my mother, who traveled a long trip from China to Canada to give help to my family after the baby was born. Without her help, I would not have been able to focus on this research during that difficult time. Also, I want to acknowledge their understanding and support for my decision on studying in Canada, which is a place so far away from my hometown, and from them.

Part of this research was presented at the Graduate Student Symposium of the 2011 Canadian Conference on Artificial Intelligence [37], and the International Workshop on Web Information Retrieval Support Systems held in conjunction with the 2011 IEEE/WIC/ACM International Conference on Web Intelligence [36].

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In 1990, Tim Berners-Lee and Robert Cailliau proposed a HyperText project called "WorldWideWeb" to utilize "HyperText...to link and access information of various kinds as a web of nodes in which the user can browse at will" [7], which marked the birth of the World Wide Web (WWW, or simply the Web). After two decades of growth, the Web has become a primary source for people to share and gather information in their everyday lives. In order to facilitate the access to information on the Web, Web search was introduced as an information retrieval tool specially designed for the Web [10], and has reached great success.

Traditionally, Web search systems are not aware of the differences between users, and provide the search results solely based on the input query. In order to enhance the accuracy and effectiveness of Web search, many researchers have explored the possibility of personalizing Web search, which tailors the Web search results to the individual users based on their interests and preferences [42]. In this chapter, the background and the motivation for research on improving Web search personalization

will be introduced. The research in this thesis is based on seminal work by Luhn regarding the identification of significant terms within English documents [39].

1.1 Web Search

In the early days of the Web, there was so little information on the Internet that a simple list of Web sites could depict a complete map of the whole Web. W3 Servers was such a list edited by Tim Berners-Lee, and from a historical snapshot of this list [73] one can see that only thirty Web sites were listed as accessible in 1992. However, in less than twenty years of development, the Web has grown to be so big that even counting the number of Web sites has become a very challenging task. According to a survey conducted by Netcraft in May 2011 [49], there were more than 324 million Web sites on the Internet at that time, and this number was growing at a rate of 15 million per month. Further, the number of Web pages on the Web is growing at an even more remarkable rate. The number of Web pages in Google's index was 26 million in 1998, and this number soon exceeded the one billion mark in 2000 [3]. Just ten years later, as of July 2010, the indexed Web is considered to contain at least 27.96 billion Web pages [17].

The incredible size and growth rate of the Web have brought to surface an information overload problem [42]: analyzing and exploring the countless resources of information on the Web have long since exceeded the information processing abilities of individual users. As such, it is now impractical for users to retrieve the information they need solely by browsing. This problem had made Web search an essential tool for people to find information among the vast resources available on the Web.

A typical search engine such as Google, Microsoft Bing, or Yahoo! asks the user to input a query and returns a ranked list in which the relevant documents are placed in the higher positions. By reducing the information resources from the whole Web to the documents provided in the results list, Web search offers an effective way to solve the information overload problem. Within a results list, a user is able to browse and retrieve the needed information from a limited number of resources. Moreover, these information resources are ranked according to their relevance to the search query, and this ranking allows a search engine to take the millions of documents that match a given query and provide just ten at a time that are presumably most relevant to the query.

However, to some extent, the information overload problem still remains in traditional Web search. One reason is that users commonly employ very short queries. By analyzing over one million Web search queries, Jansen and Spink [31] point out that the average number of terms in user-submitted queries is only 2.4. Since these short queries can be matched to many documents, it is not feasible for users to consider every document in the search results list. While some have pursued approaches for assisting users to develop better (i.e., more accurate and precise) queries [8], the primary method used by the top search engines is to improve the order of the search results such that those that are most relevant to query are placed at the top of the search results list [52]. If this is done well, it reduces the need for the searcher to delve too deeply in the search results list.

However, a difficulty with the approaches to addressing the information overload problem by improving the order of the search results is that in most cases, this order is determined by the query alone. In the absence of any additional information, the

same query provided by two different searchers will produce the same set of search results, even if the information needs of the searchers are different. For example, a sailor may conduct a Web search for "piracy" to look for information about recent events of robberies at sea, but will find that the results list contains a lot of irrelevant documents about illegal copying of software, music, or movies. On the other hand, a lawyer within a record label may input the same query to find information regarding music piracy, but will receive the exact same search results list as the sailor. In both of these cases the unwanted search results are relevant to the query, but irrelevant to the search intents of the particular user. These irrelevant documents place extra burden on the searcher to manually filtering them out, increasing the time required to examine the search results, and consequently preventing the searcher from retrieving the needed information efficiently.

1.2 Web Search Personalization

The problem described above may be called the "search intent problem", and Web search personalization is a remedy for this problem. Personalized Web search systems treat individual users differently to fit their varied information needs and provide personalized search results for each user. The necessity of Web search personalization is that different searchers may have different intents behind the same query, and they often have difficulties clearly expressing their intents by specifying their queries [69]. To make matters worse, these same searchers have a tendency to use very short queries [31] increasing the likelihood that the same query is used for different information needs. Due to these two facts, the example mentioned above would not

be a rare case, but rather a common problem that exists in the practice of Web search. Therefore, Web search personalization is necessary in order to capture each user's search intents and help them to meet the specific information seeking goals.

There are in general two kinds of Web search personalization [55]: query augmentation and result processing. In query augmentation, different users may enter the same query, but this query can be automatically modified or augmented into different variations for individual users. For example, if the user is a sailor, the query "piracy" may be augmented to "maritime piracy"; the same query might be expended to "music piracy" if it is issued by a lawyer within a record label. On the other hand, personalized Web search systems that utilize the result processing model personalize the search results instead of the search query, usually by re-ranking the search results according to the current user's interests and preferences. In this case, the same query "piracy" is accepted by the search engine, but the results about "maritime piracy" are placed at the top of the results list for the sailor, and the documents about "music piracy" will be displayed at the top for the lawyer. Although both of these approaches have merit, the focus in this research is on result processing and the personalization of the Web search results rankings.

1.3 Improving Web Search Personalization

A common approach for personalized Web search is to model a user's interests and preferences within a vector representation [23, 42]. Each dimension of these vectors represents a term and the value along a given dimension is commonly the term frequency (TF) or other related measures found within the information used to generate

the personalization vector. Such information may include Web pages [2], search results [18], or browsing histories [66]. The TF value for a given term is simply calculated by counting the number of times that this term appears in the source information. In order to avoid having the term vectors become too bloated with irrelevant information, stop word removal is often employed, whereby common terms that have no value for differentiating between good and bad documents are ignored (e.g., "a", "the", "it"). In many cases, stemming [56] is also used to reduce the number of unique terms in the term vectors.

These vectors are then used to re-rank the search results based on the similarity of the documents to the personalization vector. The assumption here is that if a term is used commonly in the vector as well as in a given document in the search results list, then that document may be important for the individual searcher. Unfortunately, this is not necessarily the case. Usually, even after stop word removal, the high-frequency terms are too common to be significant, and provide little value for describing the unique characteristics of a user's interests and preferences. As such, they may not be very helpful for assisting the users to meet their specific information seeking goals. Moreover, these common terms can easily diminish the capabilities of personalizing the search results because of the potentially ambiguous nature of such terms.

As a result, the frequency of a term may not be a good indicator of the value of that term. Classical information retrieval has made the use of term frequency-inverse document frequency (TF*IDF) [74] and its variants in order to address this problem. TF*IDF takes into account the inverse document frequency (IDF), which is the inverse proportion of documents that contain a given term to all the documents in the collection. The goal of TF*IDF is to reduce the bias towards high-frequency terms

that appear in many of the documents in the collection. However, the calculation of IDF is not always feasible within the context of personalized Web search since it requires knowledge of the distribution of terms across all documents on the Web (or at least in the Web search engine's index). A more practical way is to estimate IDF based on a subset of the Web, which could be a collection of Web documents [40], or a set of local search results returned under the current search query [32][44].

Other classical work in the field of information retrieval may be useful for improving the TF approach to personalized Web search. One such work is that by Luhn [39], in which it was suggested that given a document, it is possible to identify the significant terms just based on the term frequency calculated within that document. Luhn's suggestion was that the significance of a term follows a normal distribution placed over the terms, when they are ranked according to their frequency. Essentially, he was suggesting that mid-frequency terms are the most useful terms for representing the content of the text, rather than high-frequency terms.

Inspired by Luhn's work, a novel approach to automatically identify and re-weight significant terms in a vector-based personalization model is proposed in this research. Compared to TF*IDF, this Luhn-inspired vector re-weighting approach is more feasible because it does not require knowledge of the entire collection of documents (or at least a localized collection of documents) used to generate the personalization model, but only the information about the generated model itself. As such, the amount of information that needs to be processed as the personalization model is generated is greatly reduced. The primary contribution of this research is the application of Luhn's ideas to the domain of Web search personalization, and the development of an automatic algorithm for determining both the location and shape of the normal

distribution that produces the re-weighted vector. An evaluation of this approach using a set of ambiguous queries shows that it can indeed improve the order of the search results in comparison to the original search engine order as well as a simple TF approach to personalization, when the personalization vector is sufficiently well-trained and robust. This improvement is similar to that which can be achieved by TF*IDF, but with less information processing overhead.

1.4 Research Questions

In this thesis, the Luhn-inspired vector re-weighting approach for improving Web search personalization will be explored and the following research questions will be answered:

1. **How can Luhn's work be adapted for Web search personalization?**
   (a) **How can Luhn's work be formalized within the context of Web search personalization?**
   (b) **Is it possible to define a set of parameters for this approach that work for many different personalization profiles?**
   (c) **If not, how can the information within the Web search personalization profiles be used to tune the parameters for this approach?**
2. **What is the benefit of Luhn-inspired vector re-weighting for Web search personalization? How does this approach affect the ranking of the search results?**

In order to answer these research questions, the first step that needs to be taken is to choose a baseline Web search personalization system. This baseline system should employ a vector-based model and a TF approach to weight the terms in the vector. As a result, the baseline system will likely suffer from an over-weighting problem of the high-frequency terms, which could be adressed using the Luhn-inspired approach.

Once the baseline system is decided, the second step is to implement the approach within the framework of the baseline system. The main issues in this step are (a) transferring Luhn's idea to the context of Web search personalization and formalizing the approach, (b) identifying the parameters which can be manipulated, and (c) developing techniques to automatically tune the parameters. By addressing these issues, answers to the first part of the research questions will be provided.

As the final step, an empirical evaluation will be conducted to verify the benefit of using the Luhn-inspired vector re-weighting approach for Web search personalization, resulting in answers to address the second part of the research questions. In the course of this evaluation, the discussions will focus on the metrics used to compare performance of the baseline system and the proposed approach, the evaluation methodology for conducting the experiments, and the results and observations from the evaluation.

1.5 Organization of Thesis

The reminder of this thesis is organized as follows. Related work is discussed in Chapter 2. Details regarding the Luhn-inspired vector re-weighting approach are explained in Chapter 3. Based on the baseline system, a prototype has been implemented for

studying and evaluating the approach. The system design of this prototype is documented in Chapter 4. Chapter 5 presents the method and results of an evaluation, which was conducted to test the approach in comparison to the baseline system. The thesis concludes with a summary of the primary contributions of this work and an outline of future work in Chapter 6.

Chapter 2

Related Work

Web search personalization is a rather active field of research, and a number of systems have been built to personalize Web search using different techniques. These techniques are different on the scale of personalization (single user or group of users), the way to capture the users' interests (explicit or implicit), and the type of aid the system provides to users (query refinement or result processing). This chapter will firstly introduce a taxonomy of personalization approaches and classify the different techniques used in personalized Web search systems into the directions presented in this taxonomy. In the second part of this chapter, the focus will be on the particular class of Web search personalization that is most relevant to this research: implicit content-based personalization. Details will also be given about the baseline personalization system selected for this research. The third part of this chapter will provide an introduction to Luhn's model and its theoretical foundation in Zipf's Laws.

2.1 An Overview of Web Search Personalization

Web search personalization, as described by Keenoy and Levene, "takes keywords from the user as an expression of their information need, but also uses additional information about the user (such as their preferences, community, location or history) to assist in determining the relevance of pages" [32]. How to acquire, store, and use the "additional information about the user" is the key question in the research of Web search personalization. This section will provide an overview of the different directions researchers have pursued to answer this key question in the research area of Web search personalization. Note that one particular direction (implicit content-based personalization) is reserved for discussion in more details in the next section, as it is particularly relevant to this research.

2.1.1 A Classification for Web Search Personalization

Varied directions exist in the field of Web search personalization. In a survey paper by Micarelli et al. [42], the authors suggest that there are two major categories: collaborative-based personalization and content-based personalization. In the collaborative-based approaches, users with similar interests help each other with Web search by sharing personalized recommendations of Web pages. On the other hand, content-based approaches gather each user's personal information from hard drives, search histories, user feedback, or bookmarks, and then personalize their future searches by analyzing how the content of Web search results relates to the content of the user's personal information. This classification resonates with Zhao et al. [77], who state that personalized Web search systems can be classified according to two

dimensions: users and services. The user-oriented personalization is often called collaborative filtering, which groups users in order to recommend Web search results through peer evaluation. The service-oriented personalization is called content-based filtering, that recommends Web pages to a user by analyzing the linked content and context of Web search results.

Figure 2.1 represents the discussed classification of Web search personalization in a hierarchical structure. It is further enhanced with a new dimension, explicit or implicit, to classify personalization approaches into finer categories. In collaborative-based approaches, explicit methods ask users to directly rate or recommend search results, but implicit methods gather user ratings in a non-obtrusive manner through past queries and selected search results. In content-based approaches, explicit methods and implicit methods differ in the ways used to create user profiles. A user profile stores the information about the personal characteristics of the user, from which the system can learn the interests and preferences of the user and provide effective assistance to meet the user's information goals [23]. Explicit methods in the content-based category require users to create and maintain their own user profiles manually and explicitly for the personalization, but implicit methods automatically generate and update users' profiles by inferring their preferences from the past and current interactions with the system.

2.1.2 Collaborative-Based Personalization

The foundation of collaborative-based personalization lies in the belief that people in the same community share similar interests, thus they are likely to find the same

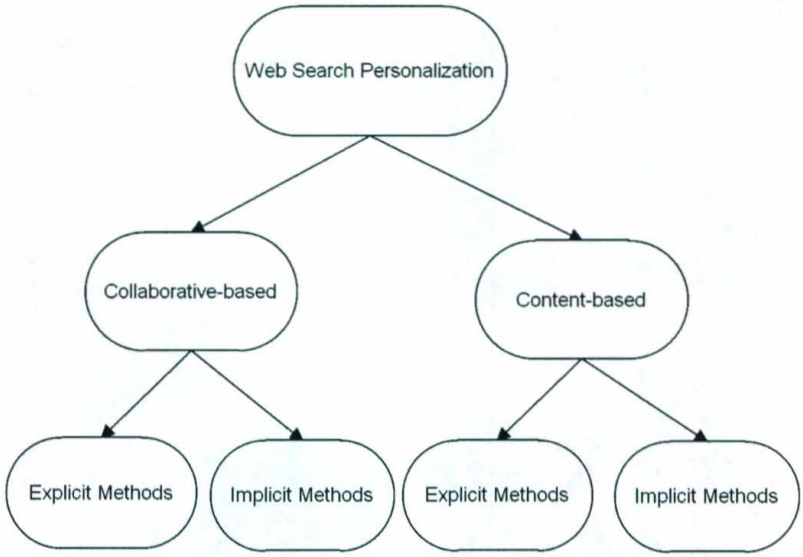


Figure 2.1: A classification for Web search personalization.

search results interesting for similar search queries. Collaborative-based approaches treat the user as a member of a community rather than as an individual, and provide the personalized search results to the user based on the recommendations from other users in the same community.

The Eurekster Siwiki [20] is an example of collaborative personalized Web search. This system allows a user to build and publish a personalized search topic. Once a search topic is published, other users who have similar interests can join this topic to conduct their own searches, and become a member of the community of this topic. Members of a topic community can vote and comment on each search result. The system learns from these explicit votes, as well as implicit clicks on search results, and adapts to the community that uses it by dynamically re-rank the search results,

pushing the most relevant results to the top of the results list.

As another instance, the I-Spy system [63] follows a similar collaborative approach, but the communities are defined as groups of visitors to different Web sites, and only the clicks on search results are used as the metric of relevance. Clicks can be considered as an implicit method of voting for search results since users will only click on the documents they think are interesting and relevant. In I-Spy, frequent visitors to a certain Web site are considered to have similar interests and so can form a user community. When these frequent visitors search within the Web site using I-Spy, a query-result relevance score is assigned to each search result based on the number of users who selected this result for the given query, and this score can be used to re-rank the search results for all users. As a result, the frequently selected search results are promoted ahead of other results if a new search for the same or a similar query is issued in the user community.

Recently, a new Wiki-like search interface was proposed by Gao and Marcos [22], which was inspired by the previous experimentations of commercial Web search providers, such as SearchWiki by Google [26] and U Rank by Microsoft [43]. The unique feature of this new search interface is that users can directly edit the ranks of the search results for a given query, and the edits can be shared among users and similar queries. Users who search for the same query are considered to share the same interests, so the rank edits could be aggregated and shared among users as user preferences. On the other hand, for a single user, the system can transfer the user's rank edits for one query to its similar ones, so the user's effort on the rank edits could be re-used.

Collaborative-based approaches could be effective since people like to accept rec-

ommendations from others who have similar interests. But the main drawback of the collaborative approaches discussed above is that the performance of personalization depends on the size and activeness of the user community. Unless a number of people are interested in a certain query or topic, and they are active enough on searching and rating this query, the collaborative search engines cannot offer the benefit of personalization to their users.

To address the problem mentioned above, some recent research has started to explore innovative ways to identify user communities. For example, Mei and Church [41] suggest that user communities for personalization could be identified based on the geographic locations of users, indicated by their IP addresses. By assigning users into nested classes based on the similarity of their IP addresses, an improvement on search results could be achieved as the result of personalization. On the other hand, Teevan et al. [72] proposed a new concept of "groupization", which is to discover and use groups of people for personalization purposes. The groups could be identified by analyzing similarities among people on query choices, personal information, and relevance judgments. An evaluation shows a significant improvement on the ranks of search results for group-relevant queries when the user's individual data is combined with the data of related people in the same group and used for personalization.

2.1.3 Explicit Content-Based Personalization

As discussed above, many collaborative approaches try to capture the common interests for a user community and recommend the relevant search results to every member of this community. For a given query, all the members within the same community

are treated the same. Everyone gets the same search results as other members. By contrast, content-based approaches try to learn personal information from each single user, and treat every user in an individual manner instead of a group manner.

A simple content-based approach is to have users explicitly describe their interests before conducting a Web search, usually by manually filling a registration form or answering a questionnaire. The data collected in this way forms the basis of the user profile, which is then used to decide which Web search results are likely to be interesting to the user by comparing the content of the search results with the content of the user's profile. For example, the now defunct Google Personalized Search [24] (which has been replaced by Google Web History) allows users to specify their interests by selecting from pre-defined topics (by clicking on checkboxes). The search results are then personalized based on a user's selection, recommending those related to the user's selected topics and filtering those that are of no interest to the user.

Explicit content-based approaches are straightforward and easy to implement, and can be effective if users can correctly create and constantly update their user profiles. Moreover, a profile explicitly created by an individual user is presumably more accurate than a profile generated through implicit methods, although White et al. [76] found that there is no statistically significant difference on the search effectiveness between implicit relevance feedback and explicit relevance feedback. Another advantage of explicit methods is that not only the positive feedback (what the user likes), but also the negative feedback (what the user dislikes) can be used in an explicit questionnaire-based approach, whereas it is difficult to infer negative feedback in implicit approaches [23].

However, it is suggested by Nielsen [51] that one should avoid requiring extra efforts from users for personalization. Moreover, a study by Teevan et al. [68] shows that people are generally unwilling to spend extra efforts to specify their intents before they search. Therefore, the effectiveness of the explicit approaches might be limited by the willingness of users to do extra work beyond entering their queries and clicking on search results that appear to be relevant. In addition, even if users are willing to fill out a questionnaire to specify their interests before search, this information may become useless as users change their interests over time. The hope that users will be motivated to constantly update the information is not valid, since they will soon realize that this is an endless burden and give it up. Moreover, when asked to fill out a questionnaire, some users may raise concerns about privacy, and feel uncomfortable submitting personal information to online services without knowing how the personal information will be stored and used.

Compared to explicit methods, a more promising approach for content-based personalization is to implicitly capture users' interests, rather than hoping users can explicitly and correctly specify their search intents beforehand. The implicit content-based approach is the direction that is the focus of this research, and will be discussed in detail in the next section.

2.2 Implicit Content-Based Personalization

A number of ways of implicitly capturing users' interests and preferences for personalization have been explored by the research community. For a detailed survey of the implicit techniques that could be used to infer users' preferences, one can refer to

Kelly and Teevan's work in [33]. Generally speaking, there are two broad categories for these implicit techniques: the ones that are based on users' Web search activities and the ones that are based on users' other activities rather than Web search. The personalization approaches that fall into the first category try to infer users' preferences on Web search results directly from their past interactions with the Web search system. For example, the submitted search queries, the selected search results, and the search logs can all be considered as implicit evidence to infer their interests and preferences. On the other hand, the approaches that fit into the second category take a broader range of a user's activities into account, and argue that this broader range provides informative context for the user's personalized Web search, which describes the user's general characteristics and interests as an individual. The typical data sources for approaches based on user context are hard drive data, browsing history, bookmarks, and emails. Sometimes the user context data could be combined with search-related information, forming a hybrid approach.

Figure 2.2 depicts the discussed categories of implicit content-based personalization approaches. In this research, the main focus will be on the approaches that are based on Web search activity. Therefore, the following parts of this section will start with a brief overview of approaches based on user context, and then the emphasis will be on the approaches based on Web search activity. The final part of this section will be a discussion of the baseline system in this research, which is an instance of the search-based approach that uses selected search results as the source of information for creating the personalization profiles.

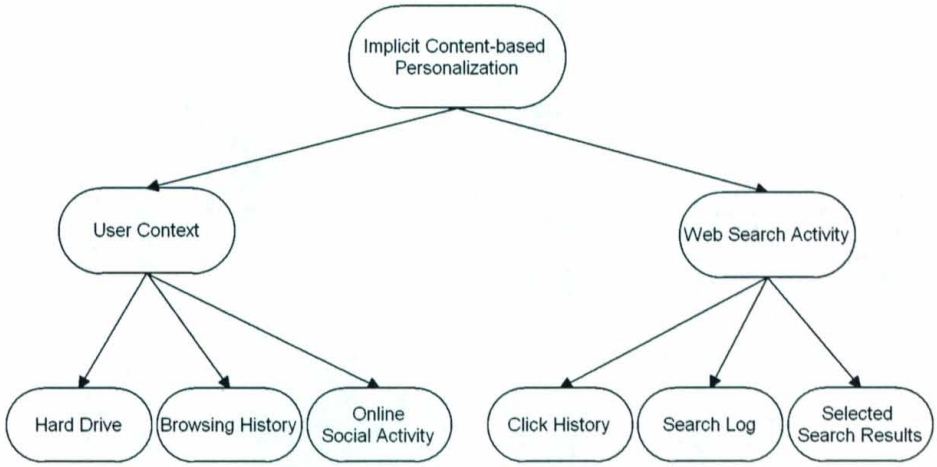


Figure 2.2: Directions in implicit content-based personalization.

2.2.1 Approaches Based on User Context

By summarizing the user's local desktop data, Chirita et al. [13] propose an approach of query expansion for Web search personalization. The "local desktop data" is defined as the set of personal documents stored on the hard drive of a user's personal computer. The useful local desktop documents are those that contain some kind of textual information, such as HTML pages, Word documents, personal textual notes, chat histories, or even meta-data of video and audio files, etc. The authors argue that these local desktop documents provide a rich repository of personal information, which could precisely describe most, if not all interests of the user. Thus the quality of the user profiles could be increased by extracting personal information from these local desktop files. For the personalized query expansion, the authors suggest three

different methods of summarizing the local desktop data. The first method is based on clustering of the entire desktop data, and selecting the query expansion terms that represent the entire desktop (which may not always be related to the user's actual query). The second method is to search for the documents that are related to the user's query on the desktop, and extract terms from the top 30 relevant documents as query expansion terms. The third method is based on the lexical dispersion hypothesis [4], wherein the query expansion keywords are acquired by extracting dispersive compounds from relevant desktop documents. According to the lexical dispersion hypothesis, those dispersive compounds can be used to represent the key concepts of the desktop documents from which they are extracted. An evaluation shows that the query expansion could indeed improve the precision of the search results, compared to the regular Google search engine.

A user's browsing history is another resource of personal information. Since such browsing history directly shows the user's interests and preferences for Web pages, the information may be more relevant to the user's Web search activity than other contextual information. Matthijs and Radlinski [40] propose a system that builds a user profile based on the complete browsing history, and then uses this profile to re-rank the search results. In this system, the user's browsing history is captured by a FireFox add-on called AlterEgo, and stored as $< URL, HTML >$ pairs. The system then extracts terms from the browsing history and weights them using three different weighting schemes: TF, TF*IDF (IDF estimates are calculated using the Google N-Gram corpus [25]), and personalized BM25 weighting [70]. A unique feature of this system is that it exploits the characteristics and structure of Web pages, and gives more weights to the terms from the important parts of the HTML documents,

such as the text contained in the title, description, or keywords tags. Also, when assigning relevant scores to the search results, the system not only considers the similarity between the search results and the user profile, but also incorporates the original ranking provided by the search engine into the final scores. Moreover, the system gives additional weights to the URLs that have been visited previously, so the user's re-finding searches could be supported as well. A user study was designed to compare the ranking results produced by this system with default Google ranking, and with two influential personalization approaches [70][18]; significant improvements were found in both cases.

Other online activities rather than browsing could also be used as user contextual information for personalized Web search. One direction is to explore the user's online social activities, as proposed by Wang and Jose [75]. In their approach, the user profiles are generated from three types of social information: blogs, social bookmarks and mutual tags. When the user issues a search query, the user's profile is used to assign an interest score to each of the search results. This interest score is combined with a relevance score provided by the underlying search engine to calculate a final score, which determines the personalized ranks of the search results. An interesting feature of this approach is that it does not stop at the point that the personalized re-ranking is generated, but further observes the user's reactions to the personalized search results. Based on the analysis of these reactions, the system adjusts the degree of personalization and the weights of different information resources, in order to adaptively develop a best setting for each individual user. An evaluation was conducted with 208 users, and the result demonstrated that the personalized search outperformed the non-personalized search for most users. Moreover, the result shows the

personalization is effective even for users who have minimal online social activities. The authors also found that utilizing multiple resources is better than using just one single resource, and the adaptive algorithm for adjusting the personalization degree and resource weights is effective.

The approach proposed by Teevan et al. [70] is considered as one of the most comprehensive and promising personalization approaches [40]. Teevan et al. employ a very rich model to represent a user, including all contextual information the user has created, viewed, or copied (e.g., Web pages the user viewed, emails the user sent or read, calendar items the user created, any textual documents stored on the user's computer). This rich user model also includes search-related information, such as the user's previously submitted search queries, and the URLs the user visited in the past. Therefore, this is indeed a hybrid approach of context-based and search-based approaches, which is making use of all available personal information on a user. The user model is then used to re-rank the top returned search results according to their relevance values to the model. In the evaluation, the authors show a significant improvement of their approach over the non-personalized ranking, and they find that merging their personalized ranking with the search engine's ranking could further improve the performance.

It is suggested [70] that the more personal data is used to represent a user, the better is the performance of the personalization. In this sense, the context-base personalization is promising because of the rich resources of the user's personal information it can employ. However, all context-based approaches share a common problem, that is they all require users to install client-side software to capture the contextual information since this kind of information can only be collected on the

client-side. This fact produces a couple of disadvantages. First, since the user profile is stored on the client side and Web search is performed on server side, inevitably there is a data exchange whenever the personalization is processed, either by transferring the user's profile to the server or by downloading a large number of search results to the client: in both cases considerable overhead costs may occur. Second, users may not be willing to install such software on their computers for the purpose of Web search personalization. Even if they are willing to do so, the distribution and maintenance of the client-side software might still be an issue. Third, users may have concerns regarding the collecting and analyzing of their personal documents on their hard drives, let alone their emails and chat conversations; these concerns may lead to rejections of the personalization system. Therefore, instead of forcing users to install client-side software, it is advantageous to collect and analyze users' information on the server side while they are doing Web searches. The personalization approaches based on Web search activities will be discussed in the next section.

2.2.2 Approaches Based on Web Search Activity

A simple approach to personalize Web search based on search activities is using click histories. A click history simply records the URLs of the selected Web pages in the search results list, without modeling them into complicated forms. PClick is such an approach proposed by Dou et al. [18]. In this approach the system records which documents have been clicked by a user for a given query. If the user issues the same query again, the previous selected documents will be pushed to the top of the results list, ranked according to the number of historical clicks on each document. In

fact, this approach is a technique to support re-finding. The motivation behind this simple approach is the observation that re-finding is common in Web search activities, and the repetition ratio in real world is noticeably high [71][67][18]. However, this approach cannot utilize personal information to facilitate searching on a brand new query because it is based on clicked documents for previously submitted queries. Also, this approach may not be very effective for searches that are repeated due to failure to find adequate information. Some may argue that if one is really looking to re-find information, they are better suited to searching or exploring within their browsing history [29]. Stamou and Ntoulas [65] provide a more complicated approach for personalized Web search based on click history, which can be viewed as a partial solution to the problems discussed above. In their approach, when a new query is issued, the system will attempt to explore the semantic similarity between this new query and query-match pages of previous queries, in order to identify the user's current preference and provide personalized re-ranking of the search results.

Another direction of analyzing Web search activities is to collect information from user's search logs. Search logs record the interactions between a user and the system, from which a large amount of the user's implicit judgments can be extracted. These judgments include user-submitted queries, selected search results, and the browsing time of each document. Using selected search results as the implicit judgment of relevance, Cui et al. [15] propose an approach for search query expansion based on extracting correlations between query terms and the document terms from a user's search logs, and then using these correlations to recommend high-quality expansion terms for new queries issued by the same user. Their evaluation showed that this approach can achieve considerable improvements in performance, especially for short

queries. Chen and Huang [12] propose another approach to personalize Web search via mining search logs. They employ not only the selected search results, but also the browsing time of each selected page to build the personalization model. This model is then used to provide personalized re-rankings of the search results. Although log-based approaches can be effective because a large amount of usage data can be straightforwardly extracted from user's search logs, a difficulty with these approaches is that they might not be very adaptable because the user's search logs may not always be available on different search engines, and the use of search logs might be restricted due to privacy concerns.

The third direction, which may have the least limitation and restriction, is to personalize a user's Web search based on the search results the user selected in the past. This approach is different from the click-history approach, although in both cases the data source is the same - the clicked search results. In click-history approaches, the system simply records clicked results as $< Query, URL >$ pairs and only supports directly matching on queries or URLs. However, the approaches in this third group usually do not directly record the clicked search results, but rather try to extract information from the set of documents the user has clicked and visited, in order to build user profiles that represent the user's interests and preferences. When used to personalize the search, these profiles are not intended to be directly matched by any search result, but rather used for computing similarities between search results and user profiles.

Dou et al. [18] propose three profile-based approaches to offer personalized re-ranking based on different lengths of the search history. The profiles employed in all three approaches are automatically generated from the clicked documents from

previous searches, and are used to calculate a personalized score for each document in the current search results. The search results are then re-ranked according to their personalized scores. An evaluation showed that both long-term and short-term search histories are important in personalized search, and the proper combination of them can be more reliable than using any of them solely.

The user profile in the work of Sugiyama et al. [66] is generated from the user's search and browsing history, which means not only are the selected search results used, but so are the Web pages that the user has browsed to by following the hyperlinks on the selected results. A weighting scheme based on TF is employed to construct the user profiles. The similarity between the profile and each document (both represented in vectors) is calculated and the search results are re-ordered based on their similarities to the profile. The authors also present a collaborative filtering algorithm as an alternate method for constructing user profiles, producing a hybrid system of collaborative-based and content-based approaches.

Ahn et al. [2] propose a profile-based personalized system for task-based information exploration. This system allows users to select and save fragments of Web pages as notes while they explored information resources. Based on the top 300 important terms judged using TF*IDF, a vector-based profile for each user is created from their notes. These profiles are used to re-rank search results using the similarity scores calculated using the BM25 formula [59] between search results and the profile. An evaluation demonstrated that this system can improve the precision of the top search results.

Based on the Open Directory Project (ODP) [50] hierarchy, Speretta and Gauch [64] create two different hierarchical profiles for each user, one generated from the

user's submitted queries and another from the user's selected search results. Both profiles can be used to re-rank the search results by combining the personalized rank with Google's rank, controlled by a tuning parameter. Evaluations showed that both profiles are equivalently effective on improving the rank of the search results. Interestingly, the evaluations also showed that the best results occur when the original search engine rankings are ignored and only the personalization profiles are used to re-rank the search results.

Similarly, Sieg et al. [62] build an ontological user profile with hierarchical concepts for each user based on the ODP; this profile is updated by a spreading activation [60] algorithm (which starts from a set of source nodes with weights or "activation" and then iteratively propagates or "spreads" that activation to other linked nodes) based on the user's interactions with the system, such as the activities of selecting or viewing new Web pages. The search results are personalized by identifying the best matching concept in the profile for each result, and using this concept to assign a ranking score to the result. An evaluation showed that the personalized search achieves 10%-25% improvement on precision over standard search.

2.2.3 Profile Models

In the profile-based approaches discussed above (including both search-based and context-based approaches that employ user profiles), two types of models are used to construct the user profiles: vector-based models [18][66][2][13][40][75][70] and hierarchical models [64][62].

Vector-based models normally represent user's interests as high dimensional vec-

tors where the term represents the dimension and the weight represents the magnitude of the vector in that dimension. On the other hand, in hierarchical models, terms are linked in a hierarchical structure, which means that two different terms may share a parent, and updates to either of the terms may also cause changes to the parent or the other term.

Compared to the vector-based model, the hierarchical model contains extra information about the relationships between terms, which can be used to provide more accurate personalized search results. Moreover, using an existing conceptual hierarchy to build the user profiles could in some degree overcome the "cold start" problem existing in personalization systems where no initial information is available in the early stages. However, hierarchical models usually start from an existing hierarchy such as the ODP [50], and this predetermined hierarchy might not always be suitable for different users with different preferences. Moreover, the construction and maintenance of hierarchical models are much more complex than vector-based models, and this fact also limits the applications of hierarchical models in personalized Web search. On the other hand, vector-based models have their own merits: vectors can easily be updated with new knowledge; they can be combined with other vectors readily; and they can be compared to one another as well as to individual document vectors produced from the search results. In this research, the focus will be on the vector-based model because of its simplicity and adaptability.

2.2.4 miSearch System

A difficulty with the discussed profile-based approaches is that most of them create a single user profile that is meant to capture all of the interests of the user. For a given information need, such a user profile will invariably include a lot of noise (things that the user is interested in for one context, but not for other contexts). The noise in the profile may cause some of search results which are irrelevant to the user's current search interests to be promoted in the search results list, and reduce the effectiveness of the personalization system.

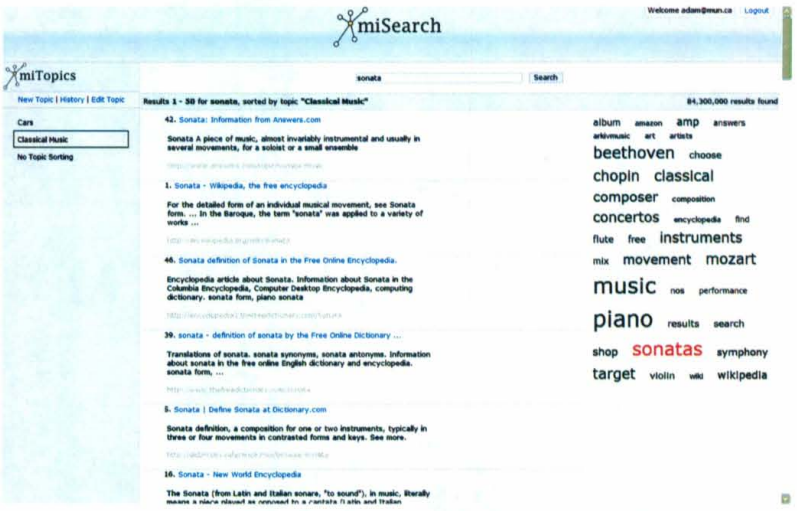
To address this problem, miSearch was developed by Hoeber and Massie [30]. Similar to other approaches based on user's interest profiles, miSearch utilizes user profiles to provide personalized re-ranking for search results. However, instead of maintaining a single profile for each user, miSearch enables every user to create multiple topic profiles. Each one of these topic profiles represents one aspect of the user's interests, and the topics can be switched from one to another as the user switches the current search goal. The main advantage of employing multiple topic profiles is that they can capture the user's different information needs separately, avoiding the noise that usually exists in a single profile, and enabling the system to offer more precise re-ranking results for the user's current search goal.

Let us look at an example of the benefit of using multiple topic profiles. Suppose there is a user of miSearch system, say Adam, who is a university professor in Music. The first topic he creates in miSearch is "Classical Music" for his work and he conducts a lot of searches under this topic for musical works like sonatas, concertos and symphonies, and musicians such as Mozart, Chopin, and Bruckner. One day, Adam

decides to look for a new car for his family, so he starts a new topic named "Cars" and searches for reviews on mid-size cars and minivans from both domestic and imported brands such as Chevrolet, Toyota, and Hyundai. While he is searching, he remembers there is a model that his neighbor recommended to him the other day, so he issues a query "Sonata" to look for the car's information and he is satisfied because he found the top search results are all about this "Hyundai Sonata" mid-size car, rather than any sonatas written by Mozart, Chopin, or Beethoven. Then, after he finishes the research on cars, he comes back to work and switches the topic to "Classical Music", under which he issues the same query "Sonata" and knows that this time he would see the information about music, not cars. If the system had only one profile to store Adam's interests, then it would be very difficult for the system to differentiate the information about cars from music, since both were highly relevant to Adam's search intents, but in different contexts. Figure 2.3 shows the screenshots from miSearch for the scenario described in this example.

In miSearch, each topic profile is represented as a term vector, which stores the information extracted from the search results clicked by the user under this topic. The fundamental premise is that the user's selection of search results provides a strong indication of relevance. While a selected document may not actually be relevant, there must have been "something" within the title, snippet, and/or URL that caused the user to think that the document might be relevant, resulting in the user clicking on it to view more details. The topic profiles are generated not to capture the relevance of an actual document, but the relevance of this "information scent" [54]. The dimensions of the profile vector are formed by the terms extracted from the title, snippet or URL of the clicked search result, and each dimension is associated with a

(a) Search results for the query "Sonata" under the topic "Cars", note that the documents about "Hyundai Sonata" are listed on the top.



(b) Search results for the query "Sonata" under the topic "Classical Music", note that the documents about sonata as a type of music are listed on the top.

Figure 2.3: Screenshots from miSearch system. Note that the search results were re-ordered according to the user's currently selected topic.

weight, which is the TF value of that term. Before the terms are added into the profile vector, stop word removal is used to ignore those terms that are most common but have little meaning, such as "the", "a", and "is". Also the terms are stemmed using Porter's stemming algorithm [56], in order to reduce terms to their root form (e.g., terms "fishing", "fished", "fish", "fisher" will be stemmed to its root term "fish"). The topic profile vector is continuously updated while the user clicks on more and more search results that he/she considers relevant under the current topic.

The miSearch system uses Yahoo! as its underlying search engine. When the user issues a query under an existing topic, the system transfers the query to Yahoo! and fetches the returned search results, and then converts each result document into vector form following a similar way of constructing the topic profile (i.e., using stop word removal, stemming, and TF weighting). The next step is to calculate the similarity between each document vector and the topic profile vector using Pearson's correlation coefficient [27]. Finally, the search results list is re-sorted in descending order based on the similarity measure, such that those documents that are most similar to the topic profile are placed at the top of the search results list.

An evaluation of miSearch was conducted based on 12 ambiguous queries selected from TREC 2005 Hard Track [45]. The result shows that miSearch can be very effective for improving the precision of the search results, even when as few as two documents have been selected by the current user.

Because miSearch uses TF in the generation of the vector-based models for the topic profiles, it may suffer from an over-weighting of the high-frequency terms. Although stop word removal is employed to address this problem, it may be possible to further improve the system performance. In this research, miSearch is employed as

the baseline personalization system for applying the Luhn-inspired vector re-weighting approach.

2.3 Luhn's Model and Zipf's Laws

As mentioned before, the approach for improving Web search personalization explored in this thesis is inspired by Luhn's model [39], which is theoretically rooted in Zipf's Laws [78]. This section will discuss both Luhn's model and Zipf's Laws, and explore the connections between these two classical pieces of research on textual information processing.

2.3.1 Luhn's Model

Luhn's model was proposed in his 1958 paper on the automatic creation of document abstracts [39]. In this paper, Luhn explains that his motivation for this work was to address two common problems existing in manual abstract processing. First, preparing abstracts is an intellectual effort that requires skill and experience. Consequently a considerable amount of manpower that could be well used in other ways is consumed on creating abstracts. Second, achieving consistence and objectivity in abstracts is difficult because the abstracts are almost always influenced by the authors' backgrounds, opinions and immediate interests; therefore the quality of abstracts may vary widely among authors. As a solution to these problems, Luhn suggests that both human effort and bias in abstract processing could be eliminated by using computers to automatically select significant sentences from the article and use these sentences to constitute the "auto-abstract".

Luhn states his basic idea as this: "It is here proposed that the frequency of word occurrence in an article furnishes a useful measurement of word significance. It is further proposed that the relative position within a sentence of words having given values of significance furnishes a useful measurement for determining the significance of sentences." [39] In other words, in order to determine which sentences may best serve as the auto-abstract, the first step is to establish a set of significant words in the article, and then use this set of significant words to identify significant sentences for constructing the auto-abstract.

In Luhn's model, judging significance of words is based on word frequency. The idea behind this is rather intuitive: a writer normally will repeat certain words while making arguments and elaborating on aspects of a subject. This means of emphasis is an indicator of significance. In practice, Luhn proposes a method for finding significant words by establishing two cut-offs on word frequency, as illustrated in Figure 2.4. The words exceeding the upper cut-off C are considered to be common words and those below the lower cut-off D are rare. Both common words and rare words cannot significantly contribute to the document's content. Luhn further proposes that the significance of words follows a normal distribution that reaches the peak at halfway between the two cut-offs, represented as curve E in Figure 2.4.

Luhn's model of word significance is unsophisticated in the sense that it avoids linguistic implications such as grammar and syntax, and also ignores the difference of word forms (i.e., words are stemmed). Also Luhn does not employ a stop-words list to remove the most frequent words like "the", "a", "is", but rather relies on the upper cut-off to exclude these noisy words, as well as those content-related high-frequency words that cannot be removed by a stop-words list, such as the word "cell" in an

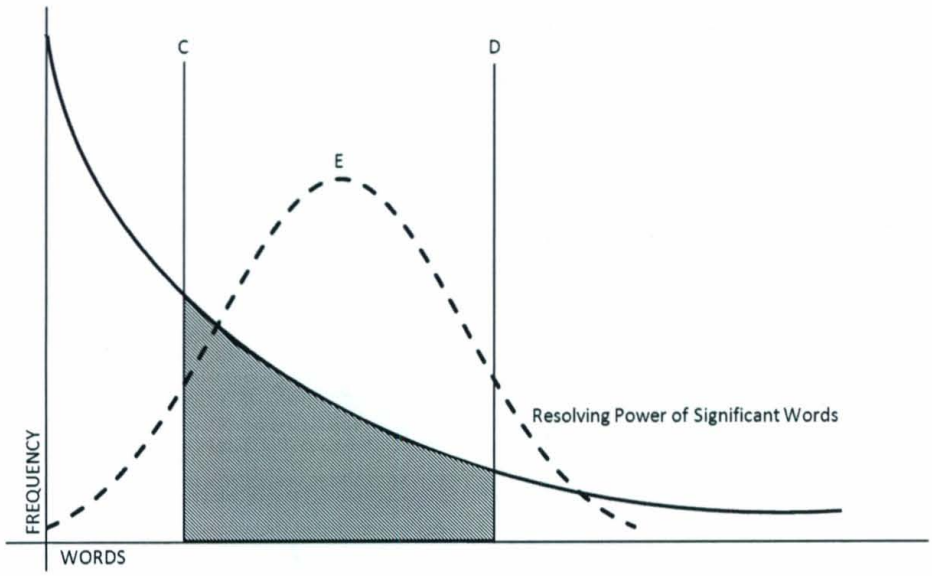


Figure 2.4: Luhn's model of word significance. This figure is adapted from Luhn's paper on word significance [39].

article on biology. In regards to the ways of establishing the upper cut-off and the lower cut-off, Luhn provides few details, only describing it as "a matter of experience with appropriately large samples of published articles".

After establishing a set of significant words, Luhn then examines the sentences according to their relationships to the significant words. The "significant factor" of a sentence is determined by the number of significant words it contains, and the linear distance between the significant words due to the intervention of non-significant words. Based on the significant factor, sentences are then ranked in descending order and the top sentences are selected as elements of the auto-abstract.

Luhn's model is seminal work in the field of automatic text processing, and forms the basis for many later works on automatic text analysis [19], automatic text sum-

marization [61], and Web search coverage testing [16]. However, to the best of our knowledge, it has not been explored in the context of personalized Web search or term vector re-weighting.

2.3.2 Zipf's Laws

Zipf's Laws [78] are classic empirical laws in linguistics. Zipf's First Law states that the frequency of a word in a given text is inversely proportional to its rank of occurrence among all words in the text. In other words, the most frequent word will occur about twice as often as the second most frequent word, three times as often as the third most frequent word, and so forth. This law could be described by the following formula:

$$r * f = c \tag{2.1}$$

where r is the rank of the word, f is the frequency of the word, and c is a constant for the given text. According to this formula, Zipf's First Law could also be stated as this: the product of the rank and the frequency of any word in a given text is approximately a constant.

Zipf's First Law is surprisingly simply yet influential. It not only holds true for most natural languages, but also has been observed in many non-linguistic contexts, such as population of cities [21], company sizes [5], and even the Internet [1]. However, in the context of word frequency, this law only holds true for high-frequency words. A reason for this is that high-frequency words tend to have unique numbers of occurrence and thus occupy unique ranks. However, many low frequency words often share the same frequency (e.g., there are many different words that appear twice or once in a

text).

In order to describe the behavior of low frequency words, Zipf also proposed a second law. Zipf's Second Law was further revised by Booth based on a more detailed analysis of low-frequency terms in text [9]. Booth's revised form of the law is

$$I_1/I_n = n(n+1)/2 \tag{2.2}$$

where I_n is the number of different words appearing n times in the text, and I_1 is the number of different words that occur only once in the text. Thus, the values of I_1/I_n, n=1, 2, 3, 4, 5, show a pattern of 1, 3, 6, 10 and 15. In other words, the distinct words that occur only once are approximately three times as many as the words appearing twice, six times as many as the words appearing three times, and so on.

2.3.3 Connections Between Luhn's Model and Zipf's Laws

When introducing Luhn's work in his classic information retrieval book [74], van Rijsbergen states that the curve of "resolving power of significant words" in Luhn's model indeed is a demonstration of Zipf's First Law, and Luhn uses this law as a "null hypothesis" to build up the two cut-offs in his model. From this statement, it is apparent that Zipf's First Law provides a start point and theoretical foundation for Luhn's model.

van Rijsbergen's statement is not the only evidence to suggest that Luhn's model is related to Zipf's Laws. Goffman's transition theory [53] provides more explanation to reveal the connections between Zipf's Laws and Luhn's model. This theory is based on the observation of Zipf's two laws: the first law describes the phenomenon of high-frequency words (i.e., unique rankings) and the second law focuses on the behavior

of low-frequency words (i.e., different words share the same frequency). These two entirely different laws predict the two distinct edges of the word distribution in any given text, and therefore it is reasonable to expect a critical region in between these two edges where the transition of word behavior from high-frequency to low-frequency phenomenon takes place. Goffman further suggests that it is at this transition region that the most content-bearing words of a given text occur, which is consistent with Luhn's model that suggests the most significant words appear in the mid-frequency region. Goffman's transition region theory provides theoretical support to Luhn's model in light of Zipf's Laws, and offers a rational explanation for the cut-offs that Luhn uses in his model to exclude both high-frequency and low-frequency words. This theory will be revisited in the next chapter as it suggests ways to automatically establish Luhn's curve of significant words in the proposed approach.

From a different perspective, Losee [38] also suggests connections between Luhn's model and Zipf's Laws. Losee argues that Luhn's model and Zipf's Laws share the same basis, which is the statistical dependencies existing between terms that can be measured by the expected mutual information measure (EMIM). Given a pair of terms, EMIM measures the amount of information that one term provides about the other. In other words, EMIM represents the "dependency" of the terms. High EMIM means the terms in the pair are highly dependent on each other, thus have to work together to express a meaning. On the other hand, a term with lower EMIM has better capability to provide information by itself, because less of the information the term carries is determined by the neighbor terms. By employing EMIM, Losee provides a partial explanation for Zipf's First Law, suggesting that the law is a consequence of the statistical dependencies that exist between terms in natural language. Also, Losee

demonstrates a phenomenon about EMIM that is consistent with Luhn's model: the EMIM decreases while moving from the most common words to the less common words, and reaches the minimal values in the region of mid-frequency terms. Once the region of mid-frequency terms is passed and one continues moving towards the low-frequency terms, the EMIM starts to increase again. This pattern of EMIM change suggests that both the rare terms and the very common terms are not as good at representing the meaning of the text on their own as mid-frequency terms, which is consistent with Luhn's suggestion.

In conclusion, van Rijsbergen's statement, Goffman's transition region theory, and Losee's EMIM-based explanations all provide indications that Luhn's model is closely connected to Zipf's Laws and thus has a sound theoretical foundation. This also explains why Luhn's model remains a practical principle in the area of information retrieval after more than 50 years. In this thesis, the possibilities of using the inspiration from Luhn to improve Web search personalization will be explored. The details of this approach will be provided in the next chapter.

Chapter 3

Luhn-Inspired Vector Re-Weighting

3.1 Inspiration from Luhn

Web search personalization has shown its effectiveness through many applications, some of which have been discussed in the previous chapter of related work. However, Web search personalization systems can be further improved by dynamically refining their personalization models to better represent users' search intents. In particular, this research focuses on the ways to refine vector-based models by identifying significant terms in the vectors and reweighting the terms according to their significance. By giving higher weights to the significant (and discriminating) terms and reducing the weights of common terms, the re-weighted personalization models can more precisely represent users' search preferences, fit their information needs, and enhance their experiences in Web search personalization.

Luhn's model [39] provides valuable inspirations for this work. Essentially, Luhn's model suggests a promising way to select significant terms from a collection of terms (i.e., a text) by analyzing the term frequency of occurrence. Similarly, the purpose of refining vector-based personalization models is to assign a significance value to each of the terms in a collection of terms (in this case, the target term vector) and re-weight the terms according to this significance value, with the same precondition that the term frequency (TF) values are given. Based on this similarity, it is reasonable to assume that Luhn's model will also work in the context of Web search personalization, although most of the existing applications of this model are for the purpose of automatic text analysis [19] and summarization [61].

3.2 Vector Re-Weighting

The first step in performing Luhn-inspired vector re-weighting is to rank the terms in the vector-based personalization model according to their frequency, resulting in a TF histogram as illustrated in Figure 3.1. In this histogram, the terms near the left end are high-frequency terms, which usually are too common to be significant. Similarly, the terms near the right end are low-frequency terms, which are too rare to be significant and can be considered noise. The valuable terms are located in the middle range.

Once the TF histogram is established, a normal distribution curve can then be placed on the top of the histogram to demonstrate the "resolving power of significant words" [39]. Luhn uses this phrase to refer to the ability of words to discriminate content: the greater the resolving power, the better the word can represent the char-

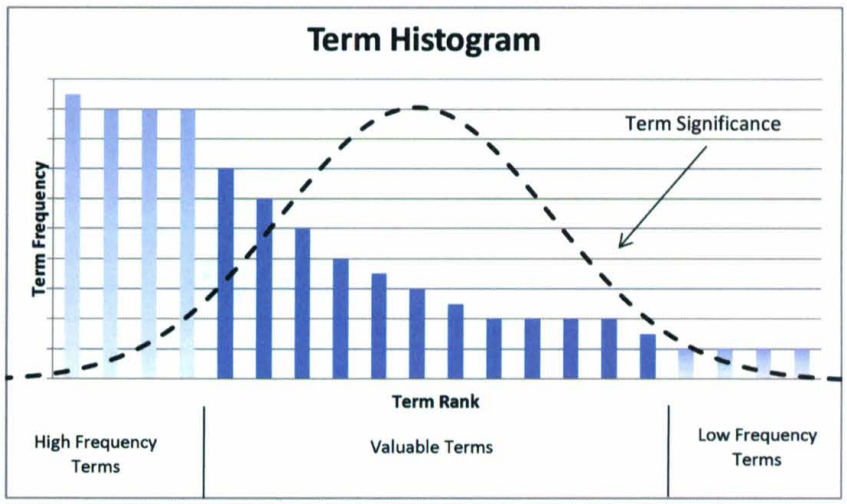


Figure 3.1: The basic idea of vector re-weighting.

acteristics of the content. As will be explained in the sections that follow, the primary challenge in applying Luhn's model is the development of an automatic algorithm to determine the location and variance of the normal distribution based on features of the term vector and its associated TF histogram.

In Luhn's original model, an upper cut-off and a lower cut-off are used to exclude the terms that provide little value in describing the text. Stop word removal is not employed in Luhn's model, so the upper cut-off plays this role instead. The lower cut-off is designed to eliminate the particularly rare terms (e.g., those that only occur once). However, as mentioned previously, the proposed vector re-weighting approach is built within the framework of an existing Web search personalization system (miSearch [30]), and this system uses stop word removal when creating topic profiles. Also, rare word removal (which removes terms that occur only once) is employed when fetching terms from topic profiles for re-weighting (i.e., those rare

terms are not considered in the process of re-weighting, but are still kept in the original profiles for data completeness). Therefore, it is not necessary to establish the two cut-offs as in Luhn's original model. In fact, since noisy terms that have no meaning have been filtered out, the remaining terms all bear some degree of value related to the search topic, and thus deserve to be considered in the re-weighting process and remain part of the re-weighted profile (although some may be assigned with very low weights after re-weighting).

There are two main reasons for selecting miSearch as the baseline system in this research of the Luhn-inspired vector re-weighting approach. First, miSearch employs vector-based personalization models based on term frequency, which is ideal for implementing and evaluating this vector re-weighting approach. Second, miSearch uses multiple topic profiles to represent a single user's different search intents, so the noise within a topic profile can be minimized. A topic profile with a low degree of noise can facilitate the study of the Luhn-inspired vector re-weighting approach. Note that while this vector re-weighting approach is developed to work within the multiple-profile framework of miSearch, it can be applied to any personalization method that employs a vector-based modeling of information that relies on term frequency.

3.3 Approach Formalization

More formally, the goal of the Luhn-inspired vector re-weighting is to replace the TF value in the source term vector with a term significance (TS) value based on the normal distribution placed over the TF histogram. The location and the variance of the normal distribution are two crucial factors for this approach. The location of the

curve determines where the peak of the resolving power of significant words is placed. In other words, it determines which terms' weights will be increased (i.e., those terms that are near the peak), and which will be decreased (i.e., those terms that are far from the peak). The variance of the curve (i.e., the degree of flatness or steepness) decides to what extent the terms in the vicinity of the peak will be re-weighted.

To calculate the TS value for each term in the vector, the probability density function of the normal distribution is used:

$$TS(i) = f(r_i; \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(r_i-\mu)^2/2\sigma^2} \tag{3.1}$$

where r_i is the rank of the given term i in the term frequency histogram, and $TS(i)$ is the significance value of that term i. Equation 3.1 contains two parameters that affect the location and the shape of the normal distribution, and therefore the TS value for a given term: the mean value μ and the variance σ^2. The mean value μ decides the location of the centre of the normal distribution, and the variance σ^2 describes how concentrated the distribution is around the mean. How these parameters for the normal distribution are determined is the fundamental challenge of making this approach an automatic method, and will be discussed in detail later in this chapter.

Once the appropriate values are determined for μ and σ^2, the TS values for the terms in a given vector-based model can simply be calculated using Equation 3.1. A re-weighted vector can then be created by replacing the frequency of term i with $TS(i)$. An alternate approach is to multiply the term frequency by the term significance, resulting in a $TF * TS$ approach for re-weighting. These two different re-weighting approaches will be compared and discussed in the evaluation chapter of this thesis.

3.4 An Example

As mentioned previously, this Luhn-inspired vector re-weighting method has been implemented within miSearch, a personalization system that uses multiple vector-based topic profiles to represent a user's various different information needs. A user may create a topic profile titled "piracy" when searching for information about old fashioned piracy: the boarding or taking control of vessels. This topic profile is populated with information as the user submits related queries and clicks on documents believed to be relevant. Essentially, this topic profile is a vector that represents the term frequency of words appearing in the title, snippet, and URL of the documents the user clicks on.

To evaluate the effect of the personalization, 50 search results were collected from Yahoo! under this ambiguous query and assigned relevance values. Based on these 50 search results, three different ranked lists were produced for comparison: the original list, the personalized list produced by miSearch, and the personalized list produced by miSearch enhanced with Luhn-inspired vector re-weighting. For this example, the mean and variance parameters of the normal distribution where manually tuned. For each of the three ranked lists, the average precision (AP) [6] was calculated over the top-10 and top-20 documents to compare the performance. A high AP value means that there are many relevant documents near the top of the list. As such, it is a measure of the quality of the ranking of the search results. Table 3.1 lists the AP values of the three ranked lists in this example.

Figure 3.2 depicts the TF histogram of the topic profile and the calculated term significance based on our approach. The blue bars in this figure show the TF data; the

Table 3.1: Average precision values for each of the three methods for ranking the search results.

AP	Yahoo!	miSearch	Luhn-inspired vector re-weighting
Top-10	0.611	0.745	0.950
Top-20	0.436	0.633	0.819

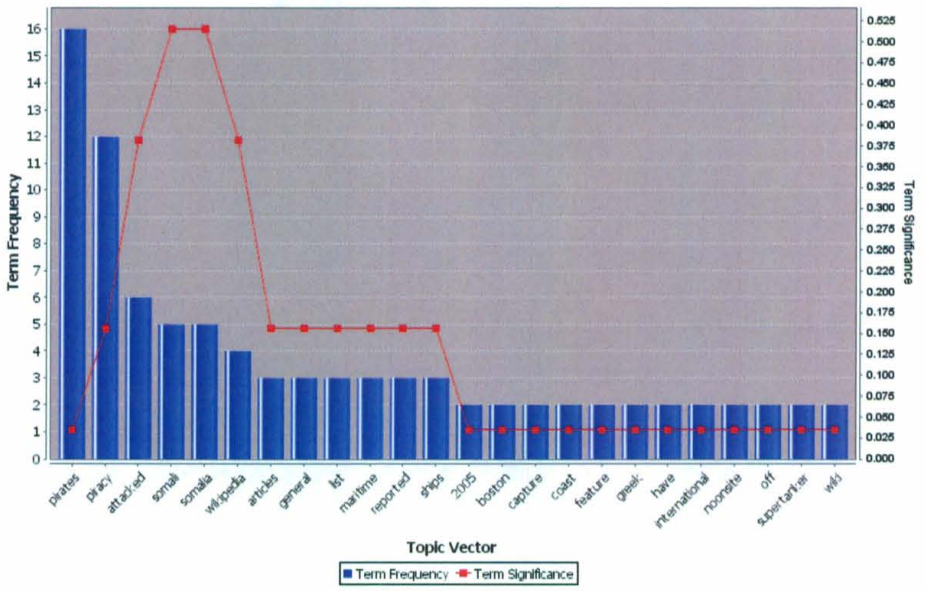


Figure 3.2: Luhn-inspired vector re-weighting for the "piracy" topic profile.

red curve represents the TS value assigned to each term by the normal distribution. Note that if two or more terms have the same TF value, they share the same rank and are therefore assigned the same TS value.

From Table 3.1, it is clear that the topic profile employed in miSearch helps

to improve the ranking order of the search results over the original search engine. The Luhn-inspired vector re-weighting further improves the ability to move relevant documents to the top of the list.

A careful inspection of Figure 3.2 can provide some insight into why this approach works. The high-frequency terms (e.g., "pirates" and "piracy") are ambiguous and may appear in documents that match many different senses of the query. However, the mid-frequency terms (e.g., "attacked", "Somali", and "Somalia") are more useful for differentiating between the desired senses of the query. As a result of re-weighting the vector, these mid-frequency terms play a more prominent role in the re-ranking of the search results.

This example demonstrates the promise of the Luhn-inspired vector re-weighting approach. However, the parameters in this example were carefully selected to ensure that the normal distribution is placed at the optimal location within the topic profile vector. In other words, this approach works well only if the parameters are properly selected. To overcome this limitation, an automatic algorithm for selecting these parameters is discussed in the next section.

3.5 Automatic Parameter Selection

3.5.1 Parameter Optimization

The first step to take in the study of automatic parameter selection is to perform parameter optimization on a set of selected queries. The purpose of this parameter optimization is to train optimum parameters for a large number of topic profiles. The

hope is that a certain set of parameters might be found to work well for many topic profiles. If this is the case, then this set of parameters can be used as the default setting for the Luhn-inspired vector re-weighting. Even if it is not the case and such a "one size fits all" setting cannot be found, it is still possible to observe some general rules for choosing parameters by analyzing the relationships between the optimum parameters and the corresponding topic profiles. These observed rules may be helpful when designing an automatic algorithm for parameter selection.

An effective technique for optimizing real number functions is Particle Swarm Optimization (PSO) [34]. PSO is a population-based, stochastic search algorithm which maintains a swarm of particles to conduct the process of optimization. Each particle contains a set of real number parameters for the objective function, and these particles move around in the search space guided by the best found position, which is continually updated as better positions (judged by the fitness evaluation) are discovered through particles' movements. In addition, each particle also has the memory of its personal best position and inertia of the last movement; the next movement of this particle is determined collectively by these three components. Consequently, all particles move toward and eventually converge at the global optimum, which represents the optimum parameters for maximizing (or minimizing) the objective function.

Ten ambiguous topics were selected from the TREC 2005 Hard Track [45] for use in these parameter optimization experiments. For each query, 50 search results were retrieved from Yahoo! and assigned relevance scores by a panel of reviewers resulting in ground truth relevance. Since the queries were ambiguous in nature, each search results list contained a mixture of relevant documents and irrelevant documents. The effectiveness of the personalization approach can be judged based on whether the

relevant documents can be identified and moved near the top of the list after the re-ranking process. Similar to the previous example, average precision measured over the top-10 and top-20 documents was used to judge the quality of the re-ranked search results lists. To facilitate the fitness evaluation, the average precision over the top-10 (AP-10) and top-20 (AP-20) documents were combined into one single fitness value by taking 60% AP-10 plus 40% AP-20.

Initially, ten empty topic profiles were created in the miSearch system, one for each of the test topics. In the second step, five searches were conducted under each topic profile, using queries that were derived from the test topic, but different than the test query. The goal here was to mimic a user's past interest and search activity in a topic. In each of these searches, the first five relevant documents were clicked to update the topic profile, and the updated profile was then used to re-rank the search results under the test query. In this way, each topic profile was updated five separate times, resulting in a total 50 different topic profiles generated for the experiments.

A test program was implemented to facilitate the experiments. Given a set of parameters, this test program automatically applies the Luhn-inspired vector re-weighting to the target topic profile and directly outputs the resulting AP values of the search results which are re-ranked by the re-weighted profile. Using this test program, it is very convenient to test a large number of different parameters, without any on-screen interaction with the system interface.

All of the 50 generated topic profiles were then used in the optimization. A set of optimum parameters was trained for each of the topic profiles using PSO with the goal of maximizing the fitness value (i.e., 60% AP-10 + 40% AP-20). Each particle in the PSO contained two parameters (i.e., μ and σ^2), and the optimum parameters

were achieved when particles converged to the global best fitness value for a given topic profile.

Analysis of the optimization results shows that the optimum parameters (normal distribution curves) vary considerably from profile to profile. As such, the conclusion is that there is no "one size fits all" set of parameters for the Luhn-inspired vector re-weighting. Therefore, it is necessary for the parameters to be determined individually according to the target profile, and this might be done by observing patterns of the optimum mean values and variances in relation to the features of the corresponding topic profiles. However, after reviewing the optimum parameters and their associated topic profiles, no patterns were observed for deciding the mean values. In other words, no general rules could be established for determining the locations of the normal distribution curves by simply examining the optimization results.

However, patterns are observed for determining the shape of the normal distribution curve. While inspecting the term frequency histograms of the topic profiles and the optimum normal distribution curves placed on top of them, a phenomenon emerges to show that the steepness of the optimum curve is related to the shape of the local region around the mean point in the term frequency histogram. A steep region in the term frequency histogram often comes with a steep normal distribution curve, and a flat region comes with a flat curve. In other words, the observed rule for deciding the variance parameter σ^2 is that if the local region around the mean point in the term frequency histogram is steep, then a low variance value (which produces a steep curve) is needed; if the local region is flat, then it is better to have a high variance value that can result in a flat curve. Figure 3.3, Figure 3.4, and Figure 3.5 contain three examples of this pattern to demonstrate how the optimum shape

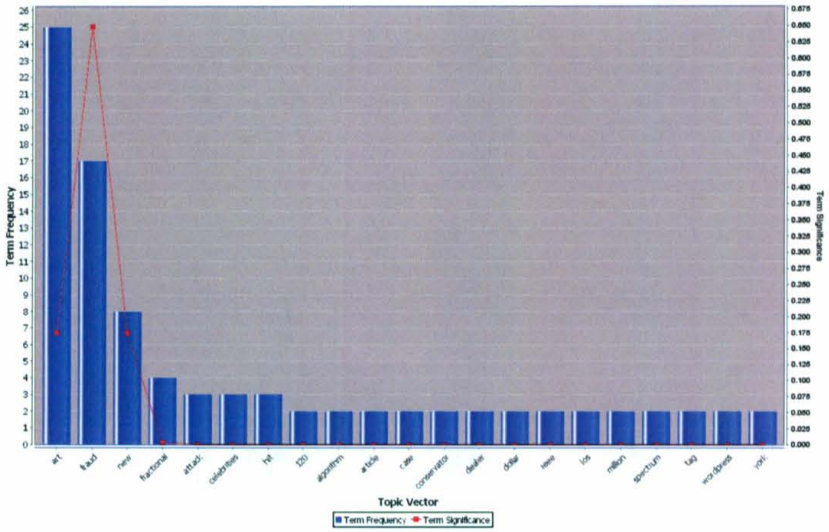


Figure 3.3: Optimum normal distribution curve for the "international art crime" topic profile ($\sigma^2 = 0.379$).

of the normal distribution curve changes from steep to flat when the corresponding histogram changes.

The optimization results provide little hints on deciding the location of the normal distribution curve, but some valuable information on deciding the shape. In the following sections, formulas for computing the location and the shape of the curve will be established, based on information gathered from this optimization experiment, along with other sources from the literature.

3.5.2 Determining the Location

The parameter optimization provides little information on choosing the mean value for the normal distribution used in the vector re-weighting process. Therefore, another

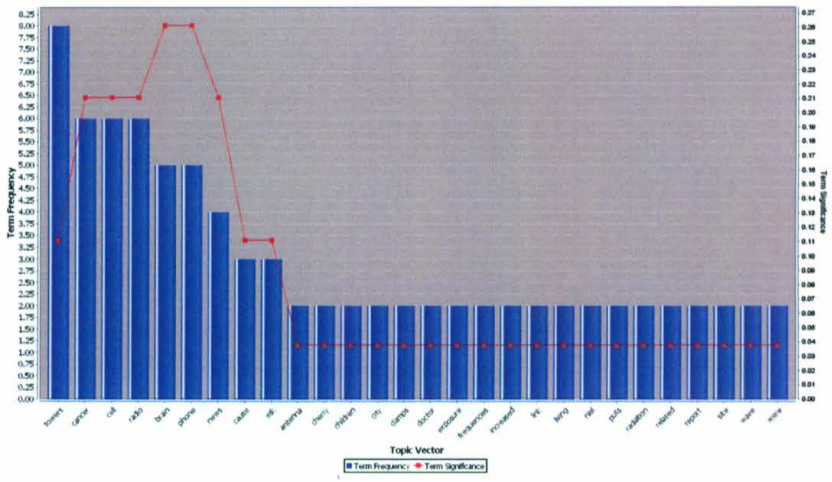


Figure 3.4: Optimum normal distribution curve for the "radio wave and brain cancer" topic profile ($\sigma^2 = 2.328$).

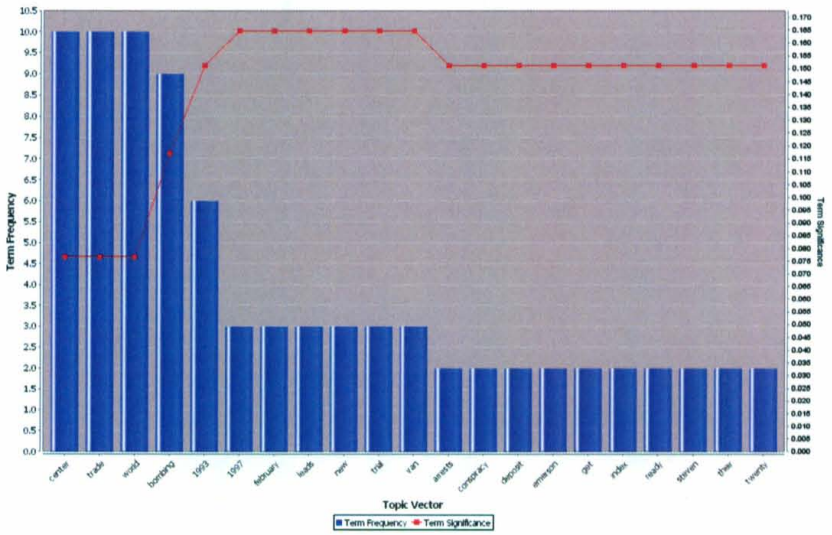


Figure 3.5: Optimum normal distribution curve for the "arrests bombing WTC" topic profile ($\sigma^2 = 5.887$).

way has to be found to determine the location of the normal distribution curve. When looking back to the root of Luhn's model, it is interesting to see that Zipf's Laws, and more importantly, Goffman's theory regarding the transition region [53], provides valuable information for this purpose.

Pao argued that the understanding of the underlying theory involved in most automatic text indexing approaches is absent, and arbitrary decisions on cutoff points are commonly made when selecting index words from frequency lists [53]. In order to address this problem, he proposed that Goffman's transition region theory, which is rooted in Zipf's Laws, provides a theoretical basis for automatic selection of indexing words.

As discussed in Section 2.3.3, Goffman's transition theory states that a region of content-bearing words could be identified for a given text. This region is situated between the high-frequency words described by Zipf's First Law [78] and the low-frequency words described by Booth's revision of Zipf's Second Law [9]. Pao formalized this theory into a simple equation for calculating Goffman's transition region:

$$n = (-1 + \sqrt{1 + I_1})/2; \tag{3.2}$$

where n is the frequency of the word that is located at the centre of Goffman's transition region, and I_1 is the number of the words that only appear once in the target text. Using this equation, one can easily identify the words around Goffman's transition region, which are considered the terms that have the highest resolving power.

Pao's equation for determining Goffman's transition region is employed to de-

cide the mean value for the normal distribution used in the Luhn-inspired vector re-weighting approach. Given a topic profile, the number of terms that occur only once (i.e., I_1) is counted within the topic profile vector, and used in Equation 3.2 to calculate the frequency value n. The term in the profile that has the nearest frequency value to n is selected as the centre of the transition region, and its rank is used as the mean value. This selected term is called the "mean term". It is possible to have multiple terms in the profile that have the same frequency value which is nearest to n, but this does not affect the mean value because these terms all share the same rank (same frequency results in same rank). Even though stop word removal is being performed in the creation of the source topic profile, this does not have an effect on the calculation of Goffman's transition region since its selection is based on selecting the term that is nearest to a calculated frequency (rather than counting within the ranked list of terms).

3.5.3 Determining the Shape

The shape of the normal distribution is an important feature in performing Luhn-inspired vector re-weighting. Within the probability density function that is used to calculate the normal distribution, the shape is controlled by the variance parameter σ^2, which is the square of the standard deviation σ of the distribution. Increasing σ makes distribution flat and broad; decreasing σ makes the distribution steep and narrow.

Whether it is better to have a flat and broad or a steep and narrow normal distribution depends upon the features of the histogram near the mean term. This

phenomenon was observed in the parameter optimization, and could be further rationalized by the fact that the mean term represents the center of the region of content-bearing terms according to Goffman's theory. Therefore, if there are many other terms near the mean term that have similar values, then a high σ value is desirable since it will flatten the distribution to include these terms that have a similar frequency (i.e., a broad transition region). On the other hand, if the terms nearby have very different TF values, then it may be better to have a low σ so that the distribution is narrowly focused on the mean term (i.e., a narrow transition region).

The slope of the histogram at the mean term can be estimated numerically using central difference formulas [11]. The 5-point central difference formula is given below:

$$s = \frac{|-f(x+2h)+8f(x+h)-8f(x-h)+f(x-2h)|}{12h} \tag{3.3}$$

where s is the approximated slope at the mean term, x is the rank of the mean term, and h is the ranking difference between any two terms on the histogram (always 1 in our case). $f(x-h)$ and $f(x+h)$ are the frequency values of the terms immediately before and after the mean term, respectively. Similarly, $f(x-2h)$ and $f(x+2h)$ indicate the frequencies of the terms located two ranks before and after the mean term.

In order to reduce the estimation error in the slope calculation, this 5-point central difference formula is used whenever possible (i.e., $x > 2$). However, it cannot be calculated if the mean term is located at the second or first terms in the TF histogram. If there is only one term located before the mean term (i.e., $x = 2$), the 3-point central difference formula [11] is used:

$$s = \frac{f(x-h)-f(x+h)}{2h} \tag{3.4}$$

For the rare case that the mean term is the first term (i.e., $x = 1$), the 2-point forward difference formula [11] is used:

$$s = \frac{f(x) - f(x+h)}{h} \tag{3.5}$$

There is a difficulty with using this numerically estimated slope within a finite calculation: it has an unlimited range of values. Since ordered histograms are always monotonically decreasing, the slope is limited to the range $(0, \infty)$. If the data in the vicinity of the mean term are very similar to one another, then the slope will approach 0; if the data in this region are very different, then the slope will be a large number (potentially approaching infinity). This range of values is hard to deal with when the slope s is used to calculate the standard deviation σ of the normal distribution.

This problem is addressed by calculating the angle of the secant line within Euclidean space. The angle θ can be calculated by the following formula:

$$\theta = arctan(s) \tag{3.6}$$

Since the slope s is within the range $(0, \infty)$, the angle θ will have a range $(0, \pi/2)$, measured in radians. That is, when the slope is near zero, the secant line will be nearly horizontal and the angle of the line will approach $\theta = 0$. As the slope of the line grows larger, the secant line gets steeper and may become nearly vertical. In this case, the angle of the line will approach $\theta = \pi/2$ (i.e., 90 degrees).

Using this angle θ, the standard deviation σ can be calculated using the following formula:

$$\sigma = a + b/\theta \tag{3.7}$$

where a and b are two tuning parameters that can be used to control the minimum value and the rate at which σ changes as a result of a change in θ. A large θ indicates

a steep slope, resulting in a low standard derivation that produces a narrow normal distribution around the mean term. A small θ indicates a flat slope, resulting in a high standard deviation that produces a broad distribution around the mean term. Note that in some rare cases, it is possible for θ to be equal to 0 and this will make Equation 3.7 illegal. In this case, σ is directly set to 10, which will result in a very flat distribution curve.

Once the standard derivation σ is generated from θ, the variance σ^2 can be easily computed by taking square of σ. This variance σ^2 could then be used, along with the mean value μ, to form the normal distribution curve that re-weights the target term vector.

As illustrated above, the only information required for performing the automatic parameter selection is the target term vector itself (note that the tuning parameters a and b are optional and could be set with default values). The features within the target vector are used to determine the mean value μ and variance σ^2 of the normal distribution curve used in the proposed Luhn-inspired vector re-weighting approach. Therefore, given a term vector, the proposed approach is able to automatically decide the re-weighting parameters and perform the re-weighting, without any extra knowledge other than the information already contained in the target vector. This feature makes the proposed approach very flexible for adapting to applications with vector-based models in different contexts, since there is no contextual information needed for conducting the re-weighting on the target vector-based models.

3.6 Discussion

Inspired by Luhn's model, a vector re-weighting approach was proposed in this research in order to improve the vector-based models used in Web search personalization systems. In this chapter, details about this approach were given to answer this research question: how can Luhn's model be adopted for Web search personalization?

The first part of this research question is about how Luhn's work can be formalized in the context of Web search personalization. In order to address this question, Luhn's model was modified and implemented through three steps: first, the two cut-offs used in Luhn's original work were discarded because of the common practice of using stop word removal in Web search personalization systems. Second, a formula based on the normal distribution density function was given to implement the vector re-weighting by placing a normal distribution curve on top of the term frequency histogram of the target vector. Third, two crucial parameters in the formula, the mean value μ and the variance σ^2, were identified as the adjustable parameters to control the performance of the re-weighting.

After the approach has been formalized, the second part of the research question asks if there is a set of parameters for this approach that could work well for many vector-based personalization profiles. This question was addressed through parameter optimization experiments, the results of which showed that the answer to this question is no. There is no "one size fits all" setting of the parameters that could be used everywhere. Instead, the parameters should be decided individually according to the features of the target personalization profile.

Although this research failed in the effort to find a global set of parameters, the

parameter optimization experiments provided valuable information on how to use the features within the term frequency histogram of the target profile to decide the shape of the normal distribution curve used in the re-weighting. Using this observation, along with Goffman's transition region theory, formulas were established to calculate the mean value μ and the variance σ^2 dynamically based on the features of the target profile. This algorithm of automatic parameter selecting provided answers to the final part of the research question about how the features within the personalization profiles could be used for tuning the parameters, and makes the proposed vector re-weighting approach flexible and easy to implement. In fact, with this automatic parameter selection algorithm, the re-weighting process can be automatically applied to the target personalization profiles to improve the personalized rankings, without any extra user effort beyond what is already required by the baseline personalization framework.

As Luhn's model succeeded in selecting significant terms for automatic abstract creation, it is reasonable to expect that this Luhn-inspired vector re-weighting approach could also achieve success in refining the vector-based personalization profiles and improving the performance of Web search personalization. The example provided in this chapter demonstrated the promises of this approach on improving the re-ranking quality of the baseline personalization system once the parameters are properly selected. Moreover, the automatic algorithm for selecting parameters was built based on the sound theoretical foundation of Goffman's transition region theory, and was designed to be tunable. There are reasons for believing that this automatic algorithm can work well and will produce high quality parameters. However, it is not expected that these system-generated parameters could match the optimum param-

eters generated one by one using PSO in the parameter optimization process (Note that this parameter optimization requires the knowledge of the relevance of documents, which is not present for generalized Web search. As such, using this approach to tune parameters is not feasible in the general case, and that is why an automatic algorithm is needed to tune the parameters based only on the features of the term frequency histogram). The goal for the system-generated parameters is that they can produce satisfactory re-weighting results based solely on the topic profiles, rather than on supplemental information that may or may not be reasonable to collect.

Chapter 4

Prototype Implementation

In this chapter, a prototype for studying and evaluating the Luhn-inspired vector re-weighting is presented. The design, architecture, and user interface of this prototype will be discussed in the following sections.

4.1 System Design

Based on the miSearch system [30], a prototype has been built in order to study the Luhn-inspired vector re-weighting approach and to conduct evaluation experiments. Specific prototype design goals are listed as follow:

- Focus on studying the approach. The prototype was built in the early stage of this research and continually updated while the research progressed. One main function of this prototype is to implement new ideas and verify their outcomes, especially in the study of the automatic parameter selection. Therefore, the prototype was designed to make it easy to manipulate the re-weighting parameters,

visualize the re-weighting results, and output the judgments of the re-ranking quality.

- Facilitate the evaluation. For a large-scale evaluation, it is not feasible to prepare test data and conduct experiments merely through manual interactions on interfaces. The prototype provides functions to automatically (or semi-automatically) generate the test data, run the experiments, and output the evaluation results.

- Add TF*IDF feature. The original miSearch system only employs a TF scheme for modeling the personalization profiles, and thus could be further improved by adding a TF*IDF based approach on profile modeling. Moreover, the evaluation can be more comprehensive and valuable if the proposed re-weighting approach is compared not only to the TF scheme, but also to the TF*IDF scheme.

- Do not break the original architecture and workflow of miSearch system. The vector re-weighting and TF*IDF are designed as add-on features to the original miSearch system. The method for keeping the topic profiles independent of one another and for collecting the information for populating the topic profile based on clicked search results remains unchanged within the prototype.

- Light-weight implementation. The prototype minimizes the modification to the original miSearch system, and reuses the existing code as much as possible.

- User-friendly interface. In this prototype, the user interface is not designed for the end users (it is too complicated for the end users), but for the researcher to study the approach. The purpose is to support the researcher to switch between

different approaches and manipulate the parameters conveniently.

4.2 Architecture

4.2.1 Platform

The prototype is integrated into the miSearch system, which is a standard Java Web application built on Java Development Kit 1.6. The system is developed using NetBeans as the integrated development environment, and GlassFish as the Web application server. MySQL is used to manage the topic profile database, as well as the user accounts.

4.2.2 System Architecture

The system architecture of the prototype is shown in Figure 4.1. Since this prototype is built on the existing miSearch system, this architecture includes the structure of the original miSearch system, as well as the new modules developed specifically for this research. These two parts are distinguished by different colors. The general workflow and the relationship between modules will be outlined here; more details about the design and implementation of the new modules will be given in the next sections.

After *logging* into the system, the user can create a new *search topic* or select an existing one and issue a *search query*. This query is passed to the *search results generator*, which generates a ranked list of search results using the *Yahoo! search engine*. If the search query is a new query, the search results generator also stores the first page of the returned search results into a local *search results cache*, so the

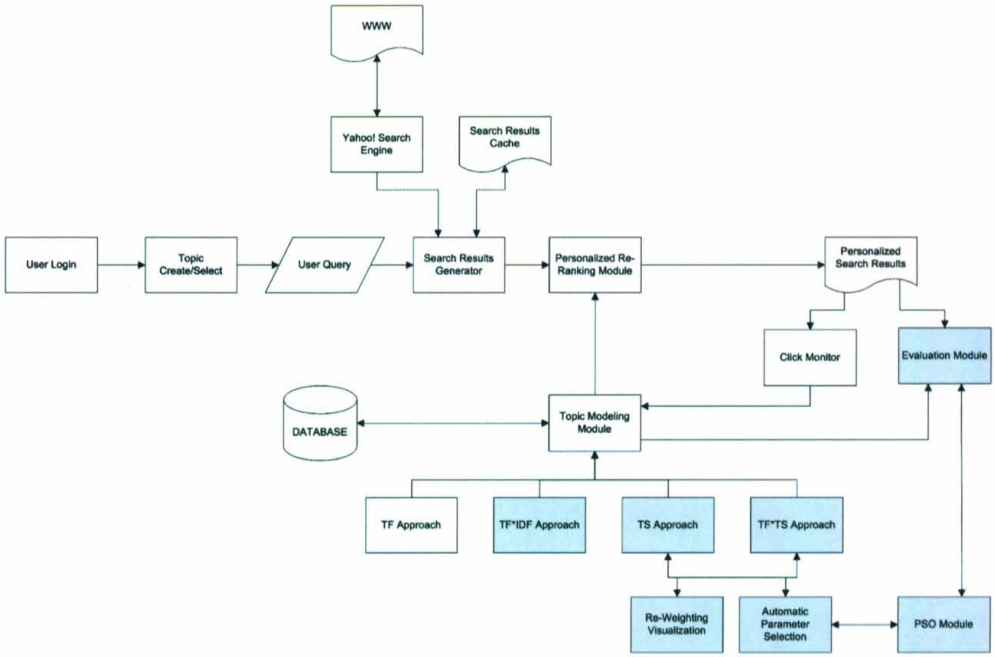


Figure 4.1: System architecture of the prototype. The modules marked in blue color are new modules developed for this research.

generator can directly fetch the search results from the cache if the same query is issued again. The search results are then sent to the *personalized re-ranking module*, which re-ranks the search results according to the similarities between the search results and the current topic profile. Finally, the re-ranking results are presented in a re-ranked list of *personalized search results*.

The heart of the prototype is the *topic modeling module*. This module models topic profiles that represent the user's interests and preferences within specified search topics; these topic profiles are used to determine how the search results should be re-ranked. A *click monitor* watches the user's behavior of clicking the documents in the

results list, and exacts information from the title, snippet, and URL of the clicked documents to generate/update the topic profiles. The topic profiles can be modeled using different approaches. The *TF approach* is used in the original miSearch system; the *TF*IDF, TS,* and *TF*TS approaches* are newly added to this prototype for this research. More details about the new approaches will be given in the following subsections.

The *PSO module* and the *evaluation module* are designed not as features of the prototype, but as tools to facilitate this research. The main functions of these two modules will be introduced later in this section.

4.2.3 TF*IDF Approach

TF*IDF [74] is a classical weighting scheme in information retrieval. For each term t in a given document d, the TF*IDF weight is calculated by multiplying the frequency of the term t in document d by the inverse document frequency (IDF) of t:

$$TF * IDF(t, d) = TF(t, d) * IDF(t) \tag{4.1}$$

The IDF aspect of this formula is calculated as follows:

$$IDF(t) = log\frac{|D|}{DT(t)} \tag{4.2}$$

where $|D|$ is the total number of documents in the corpus from which the document d is retrieved. $DT(t)$ is the number of documents in the corpus D in which the term t appears. Essentially, TF*IDF gives high weights to the terms that have a high term frequency (in the given document) and low document frequency in the whole collection of documents, and low weights to the common terms that appears in many documents.

Applying TF*IDF in the context of Web search personalization is always a challenge. The main difficulty is that the calculation of IDF requires knowledge about the entire corpus of the documents, and this corpus is the entire Web in the context of Web search personalization. While Web search engines likely have access to this information, they do not make it available through their search APIs. As such, for a search mechanism built upon the API of the top search engines, it is not feasible to calculate a global IDF for Web search results using the entire Web because of its incredible size. While some may argue that it is possible to conduct a separate Web search for each term or stem that makes up the personalization vector or document vector to get the document frequency, these vectors may have a very high dimensionality making this approach rather expensive with regards to network resources, Web search service resources, and the time taken to make the calculations.

Another approach is to estimate IDF using a subset of the Web. This subset could be an existing collection of Web documents as used in [40], or a set of search results retrieved by the current query [32][44]. A collection of Web documents could better mimic the size of the Web, but it might be out of date and may not provide an accurate sample of the current Web. On the other hand, using the current search results as the corpus can avoid the extra efforts on accessing and maintaining an external collection, and makes use of the search results data that is already at hand.

In this prototype, a subset of current search results is used to calculate the TF*IDF. When the *click monitor* notices a document in the search results list has been clicked, it passes this document, along with the current page of search results (100 search results per page by default), to the *TF*IDF approach module* via the *topic modeling module*. The terms in the clicked document are extracted and TF

values for the terms are calculated. Next, for each term, the number of documents that contain this term is counted by searching the term in the collection of documents (i.e., 100 documents in the current page of search results). Note that by using more documents, the accuracy in estimating IDF can be increased, but the trade-off is that more information has to be processed. The results are then stored into the database as the TF*IDF profile in the format of $< Term, TF, DT, |D| >$, in order to keep the raw data for calculating TF*IDF available. This is done since two TF*IDF vectors cannot be simply added when new documents are clicked. In the case of updating the TF*IDF profile, the TF, DT, and $|D|$ values for the same term in the original profile and in the newly clicked document are added respectively. These updated values are then used to re-calculate the new TF*IDF weights for the terms in the profile using Equation 4.1 and 4.2.

When the user chooses to re-rank the search results based on TF*IDF, the TF*IDF profile is passed to the *personalized re-ranking module*. This module converts the documents in the results list into TF*IDF document vectors (note that it is different from TF, TS, and TF*TS approaches, where the results document are converted into TF vectors) following a similar procedure as generating the TF*IDF vector from the clicked document, but without the step of storing the raw data into the database (since the document vectors will not be updated). Each of the document vectors is then compared to the profile vector and a similarity score is calculated using Pearson's correlation coefficient [27]. Based on the similarity scores, the documents in the search results list are re-ordered in a descending order of similarity, resulting in a personalized search results list based on TF*IDF modeling.

4.2.4 TS Approach

As described in Chapter 3, The *TS approach* is one of the two ways to model topic profiles using Luhn-inspired vector re-weighting. Two re-weighting parameters (μ and σ^2) are required for this approach to perform the re-weighting. These parameters can be input manually, or generated automatically by the *automatic parameter selection module* given the tuning parameters (a and b). In both cases, the approach talks to the database first to fetch the current TF profile, and then re-weights the terms in the profile vector using the term significant (TS) values calculated based on the re-weighting parameters and the features of the TF histogram generated from the TF profile. The re-weighted topic profile is then sent to the *personalized re-ranking module*, which converts the result documents into TF vectors and compares them to the re-weighted profile (TS vector). Note that another alternative would be to re-weight each document vector using this same Luhn-inspired approach; however, it is expected that due to the sparseness of these vectors, the accuracy in choosing the location and shape of the normal distribution may not be as good as with the more robust topic profile vectors. Next, similarity scores between the profile vector and document vectors are calculated using Pearson's correlation coefficient [27]. According to the similarity scores, the search results are re-ranked to form the personalized search results list.

For the TS approach, the results of the re-weighting can be visualized through the *re-weighting visualization module*. Both the original profile and the re-weighted profile are passed to this module. Using the TF values in the original profile, the re-weighting visualization module generates a term frequency histogram where the terms

are ranked in descending order of term frequency. Based on the re-weighted profile, a normal distribution curve is placed on the top of this term frequency histogram, which represents the re-weighting results of the profile vector. The visualization offers an intuitive way to look into the mechanism of the re-weighting in action, and provides a clear picture to show how the re-weighting helps to improve the original profile. Moreover, through the visualization, it is straightforward to see the changes to the re-weighting results while tuning the parameters, which provides valuable information for studying the parameters in this research.

4.2.5 TF*TS Approach

*TF*TS approach* is an alternative method to refine topic profiles using Luhn-inspired vector re-weighting. The re-weighing process is almost identical to the TS approach, except that the new weights assigned to the terms in the profile vector are calculated using the product of term frequency (TF) and term significance (TS), resulting in a TF*TS scheme instead of using TS alone. In this case, the re-weighted profile is a TF*TS vector, and personalized ranking results can be produced via similarity calculation between this TF*TS vector and the TF document vectors.

Similar to the TS approach, the TF*TS approach can be visualized via the *re-weighting visualization module*. Compared to the TS approach, the visualization of TF*TS may have a different normal distribution curve placed on the same term frequency histogram, representing the weights assigned to the profile vector by the TF*TS approach.

4.2.6 PSO Module

As mentioned in Chapter 3, the *PSO module* is mainly used to optimize the re-weighting parameters for the study of the automatic parameter selection, but it is also employed as a tool to optimize the tuning parameters in the evaluation for this research. More details about the role that PSO plays in the evaluation will be provided in the next chapter.

4.2.7 Evaluation Module

The *evaluation module* helps to prepare the test data, conduct the evaluation experiments, and organize the evaluation results. In this module, a cache-generating program converts the documents in the test data collection into the local cache format that can be recognized by the miSearch system. An evaluation program that accesses the search result documents in the cache and topic profiles in the database conducts the experiments automatically and outputs the evaluation results in metrics of precision and average precision. A Microsoft Excel template takes the evaluation results and organizes them into tables and charts dynamically, providing intuitive views of the evaluation results.

4.3 User Interface

As a typical Web application, the prototype employs an interface that interacts with the user via Web pages that are generated by the server and viewed via a Web browser. The Web-based interface is constructed using standard JSP and JavaScript techniques, and is designed to provide convenient and informative interactions to the

researcher (and perhaps expert users). In the next sections, the user interface of the prototype will be introduced in detail, and the focus will be on the new features developed in this research.

As previously mentioned, the user interface of this prototype is not intended to be used by the end users, but by the researcher in this study. As will be illustrated in the following sections, many features of the interface are designed for experimental purposes: the approaches for modeling the topic profile can be switched from one to another; the new approaches can be turned on and off; parameters can be adjusted and manipulated; evaluation information can be displayed. If the end users are given this level of control, it would be too complicated for them to understand and use this system. Therefore, in the following sections, the term "researcher" will be used to refer to the user of this prototype, in order to avoid misunderstandings.

4.3.1 The Main View

The main view of the prototype is shown in Figure 4.2. There are three main components in this view: the left panel provides main functionality for selecting, managing and re-weighting user-defined search topics, and a switch for choosing the re-ranking and re-weighting modules. The centre panel is the place where the researcher can input the search query, conduct the search, and review the search results. The right panel offers a tag cloud to visualize the selected topic profile, and a re-ranking evaluation tool to display the precision and average precision values calculated based on the top-10 to top-50 search results.

In this main view, the researcher can select an existing search topic and conduct

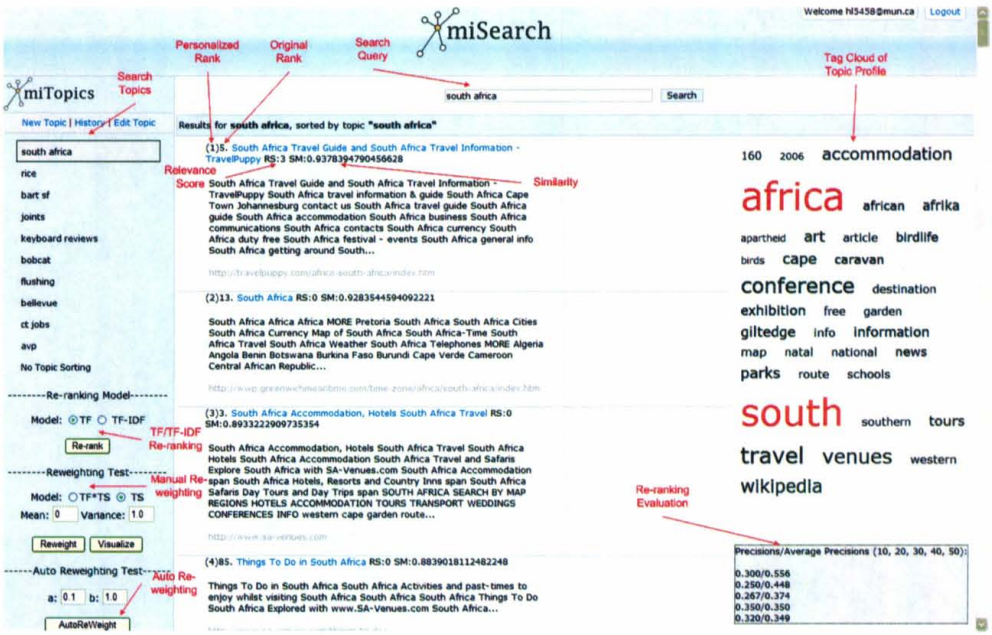


Figure 4.2: The main view of the prototype.

related searches, or create new topics and issue new queries. In either case, the topic profile will be populated while the researcher clicks on the documents that are considered relevant in the search results list. The changes of the profile will be visualized by the tag cloud, and will affect the ranking of the search results once the page is refreshed. The researcher can also try the new features that are listed in the left side of the view, to re-weight the topic profile for better ranking results, or choose the model used to re-rank the documents. Note that the display of the relevance score, similarity, and re-ranking evaluation in this view is for research purposes and could be turned off when the system is used by a real end user.

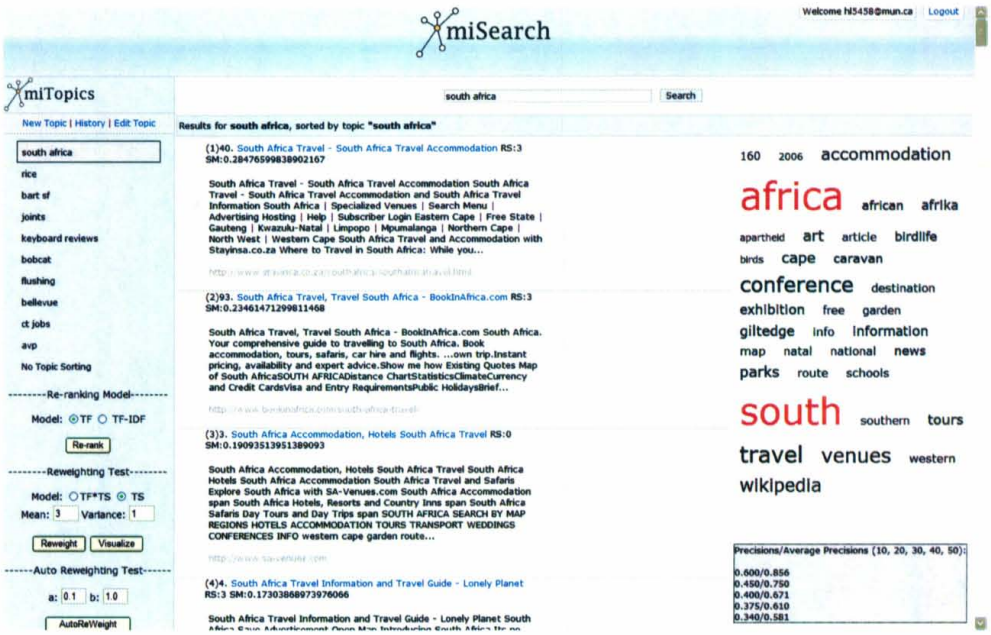


Figure 4.3: The view of manual re-weighting.

4.3.2 Manual Re-Weighting

From the main view shown in Figure 4.2, the researcher can test the Luhn-inspired vector re-weighting feature by manually inputting the re-weighting parameters (mean value μ and variance σ^2) and clicking on the "Reweight" button. Figure 4.3 shows the view after manual re-weighting is performed. Note that the documents are re-ranked and the precision and average precision values are changed in this view.

The researcher can also visualize the re-weighting by clicking on the "Visualize" button. The re-weighting parameters are still required in this case. As mentioned previously, the re-weighting visualization module provides a term frequency histogram to visualize the topic profile, and a normal distribution curve that represents the re-weighting result. An example of the visualization is given in Figure 4.4.

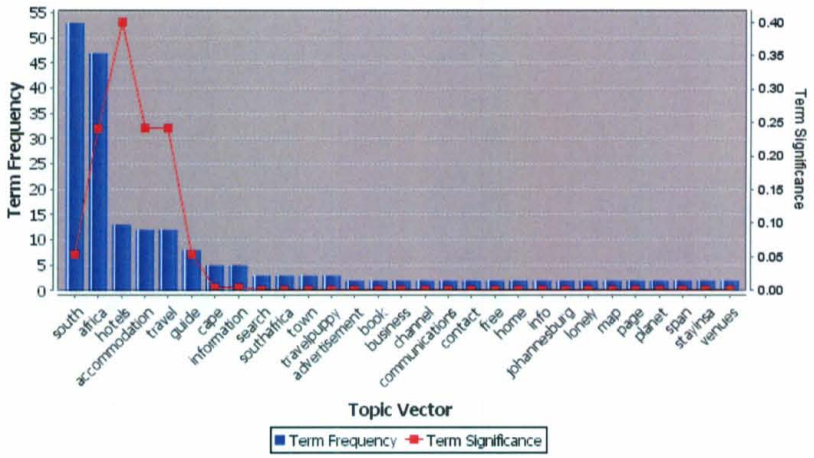


Figure 4.4: An example of re-weighting visualization.

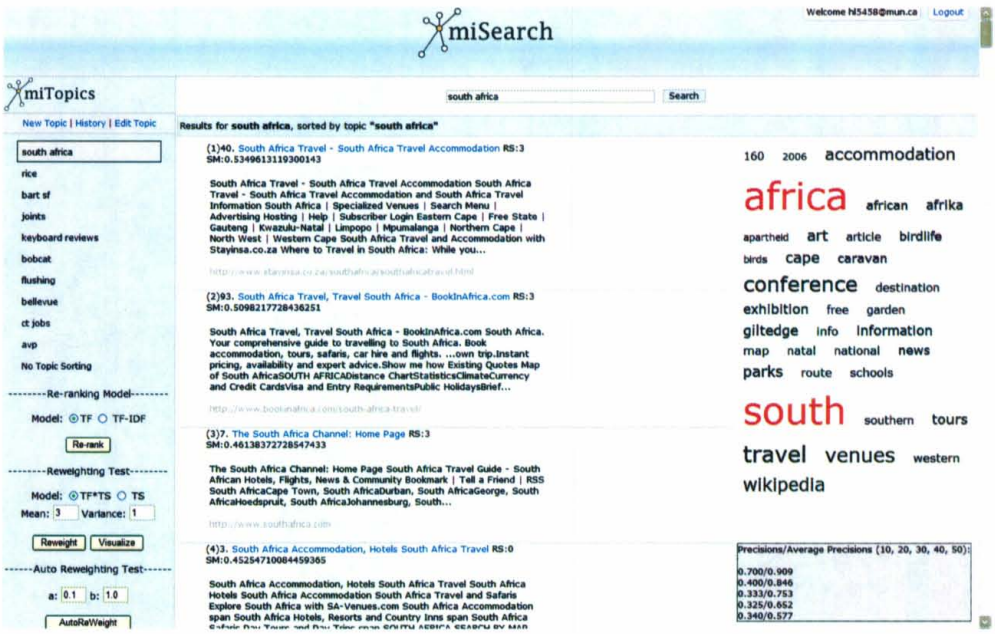


Figure 4.5: The view of TF*TS re-weighting.

In the manual re-weighting, TS is used as the default re-weighting method. However, the researcher could conveniently switch to TF*TS method by selecting it in the radio button. Once the TF*TS method is selected, a click on the "Reweight" button will produce a ranked list of search results based on the TF*TS re-weighted topic profile, and the visualization will also be changed accordingly. Figure 4.5 demonstrates how the ranking of documents is changed when the TF*TS method is employed, even with the same re-weighting parameters.

4.3.3 Automatic Re-Weighting

Besides re-weighting the topic profile manually, another option is to let the system automatically generate the re-weighting parameters. The researcher may try to tune the algorithm using the two tuning parameters, or simply keep the default tuning values ($a = 0.1$ and $b = 1$) and click on the "AutoReWeight" button to view the re-weighting results based on the automatically generated parameters. Figure 4.6 shows the view of the automatic re-weighting.

After the automatic re-weighting is performed, the parameters generated by the system are automatically input into the manual re-weighting control panel. Therefore, the researcher can clearly know what the system-generated parameters are, and have the opportunity to further tune these parameters to re-weight the profile. Also, the researcher can visualize the automatic re-weighting result by clicking on the "Visualize" button using system-generated parameters.

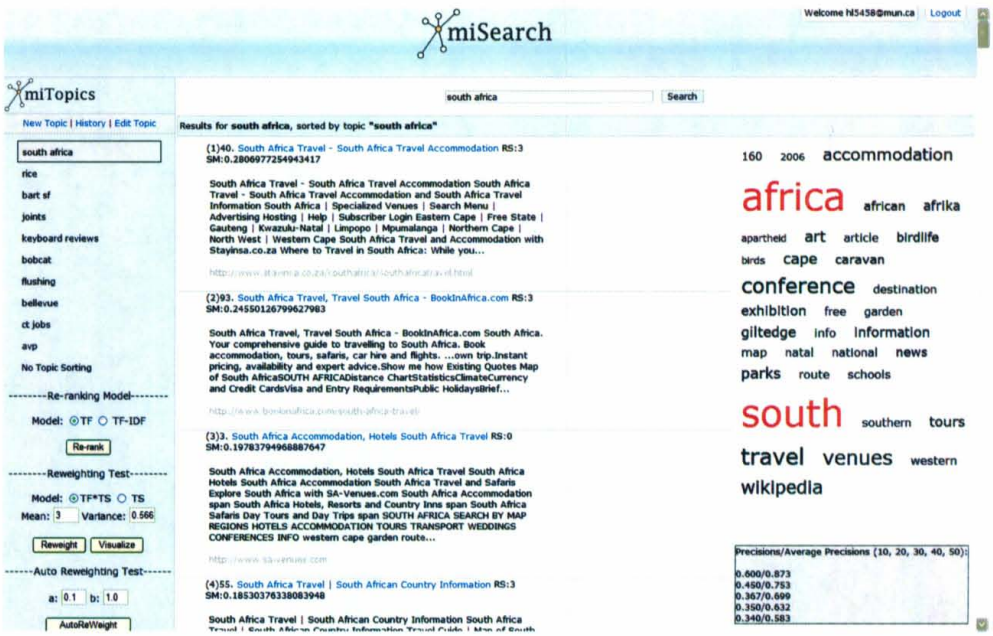


Figure 4.6: The view of automatic re-weighting.

4.3.4 TF*IDF Re-Ranking

As an option, the researcher may choose to re-rank the search results using TF*IDF modeling. In order to do so, one can simply select TF*IDF in the radio button and click on "Re-rank" button. The re-ranking results are shown in Figure 4.7.

In Figure 4.7, note that the search results are ranked by TF*IDF in a different order compared to the TF re-ranking shown in Figure 4.2, and thus produce different precision and average precision values. In addition, when the TF*IDF modeling is selected, the features that only work for TF modeling, such as the profile re-weighting and the tag cloud visualization, are disabled and hidden in the interface. This results in a very different view, which provides strong cues for the researcher to distinguish the TF*IDF re-ranking from the default TF re-ranking.

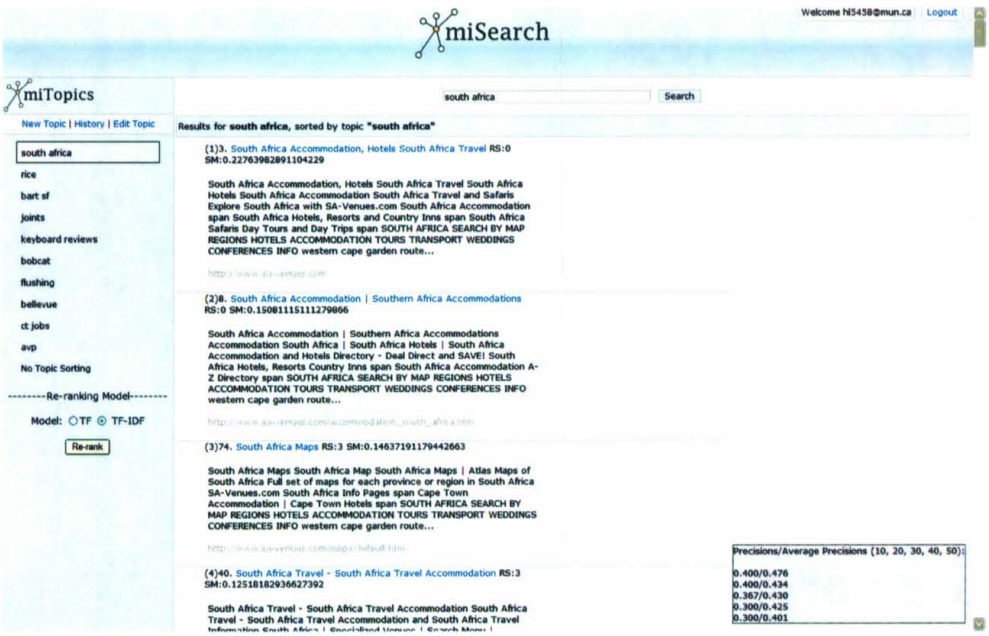


Figure 4.7: The view of TF*IDF re-ranking.

4.4 Discussion

In this chapter, the design, architecture, and user interface of a prototype for studying the Luhn-inspired vector re-weighting are presented. This prototype is implemented based on the existing miSearch system, and benefits the study and the evaluation of the proposed approach in this research.

The prototype is a faithful implementation of the proposed Luhn-inspired vector re-weighting. As proposed, the re-weighting parameters play a key role in both of the manual re-weighting feature and the automatic re-weighting feature to determine the re-ranking results. Moreover, the correctness of the implementation of the re-weighting algorithm and the automatic parameter selection algorithm can be ver-

ified by the re-weighting visualization and the explicitly displayed system-generated parameters.

The prototype helps the study of the proposed approach. The real time precision and average precision calculation provides a quick judgment of the re-weighting results, and the re-weighting visualization unveils full details of the re-weighting mechanism. With these two handy tools, it is very convenient to manipulate the parameters to see the changes through manual re-weighting or automatic re-weighting, and more importantly, understand why and how these changes of the parameters affect the re-weighting results.

The prototype helps the evaluation of the proposed approach. Through the dedicatedly designed evaluation module, the prototype offers useful tools for preparing, conducting, and organizing the evaluation experiments. These tools ease the workload of the experiments, and allow a large-scale evaluation of the proposed approach to be possible. Also, the newly added TF*IDF modeling offers a new dimension for comparing the re-ranking results in the evaluation, thus making the evaluation more comprehensive.

The implementation also meets other design goals. The new features are integrated into the existing miSearch system in a non-intrusive manner, and the newly designed user interface is helpful for assisting the researcher to experiment with the new features and manipulate the parameters.

In conclusion, the implementation of the prototype verifies the feasibility of the proposed Luhn-inspired vector re-weighting approach and provides new features that enhance the existing miSearch system. More importantly, this prototype helps to answer the research questions by providing valuable assistance for the study and

evaluation of the proposed approach.

Chapter 5

Evaluation

An experimental study was conducted to evaluate the effectiveness of the Luhn-inspired vector re-weighting approach. The methodology, settings, and results of this evaluation are presented in this chapter. The key question to be answered through this study is whether the proposed approach is indeed an improvement over the ranking order of the original search results, the existing baseline TF approach, and the commonly used TF*IDF approach.

5.1 Methodology

In the evaluation, the main goal is to compare the ranking orders of search results produced by a search engine, to that of the miSearch system (TF approach), the TF*IDF approach, and the Luhn-inspired vector re-weighting approach proposed in this thesis. In order to do so, it is necessary to have a dataset which contains a corpus of documents, a query set that defines queries for the search engine to generate search results from the data set, and relevance judgment scores for each search result to

determine whether it is relevant or irrelevant to each query.

Since miSearch employs Yahoo! as its default search engine, one possible solution is to choose a set of queries to perform searches on the Yahoo! engine, and have a panel of reviewers assign relevance scores to the search results, as in [30]. However, it is not feasible to assign relevance scores in this way when the number of queries and the number of documents retrieved under each query exceed the abilities and time limitations of the reviewers. Moreover, if others want to replicate the experiment with the same queries, it may not be possible to reproduce the exact evaluation results for two reasons: first, Yahoo! is a live search engine that will produce different search results over time as the Web is constantly changing. Second, the relevance scores provided by a different panel of reviewers for a different set of search results may not be consistent with the first.

In order to address the difficulties mentioned above, a well-recognized test collection was used to conduct the evaluation. A test collection consists of a large collection of documents, a set of queries, and corresponding relevance judgments made by a group of experts for the documents that relate to each of the queries. For this evaluation, the test queries can be selected from the provided query set, and used to conduct searches on the document corpus. In this way, relevance scores for the search results can be obtained easily and objectively from the relevance judgments provided by the test collection. Employing such test collections is a common evaluation methodology within information retrieval research, since it makes the evaluation reproducible due to the fact that the document collection, query set, and relevance judgments are published, fixed, and accessible to the research community.

Every year, the Text REtrieval Conference (TREC) [47] provides large-scale test

collections for the information retrieval community, in the form of TREC Tracks [48]. The TREC 2010 Web Track [46] is one such track provided in 2010 that focused on evaluating Web retrieval technologies. Since it fits well into the purpose and context of this thesis, this test collection was employed to conduct experiments in the evaluation.

The dataset used in the TREC 2010 Web Track is the ClueWeb09 Dataset [35]. This dataset consists of 1 billion Web pages, in ten languages, collected in January and February 2009. Since this research does not address issues with multi-lingual Web search, a subset of this dataset was used, namely TREC Category B, which contains 50 million English Web pages.

The TREC 2010 Web Track provided 50 queries for the competition. According to the official guideline [14], queries are categorized as either *ambiguous* or *faceted*. Ambiguous queries have multiple distinct interpretations, and a user interested in one interpretation would not be interested in the others. On the other hand, faceted queries have different aspects covered by the subtopics, and a user interested in one aspect may still be interested in others. Since disambiguation is the main power of Web search personalization, only these ambiguous test queries were selected for use in this evaluation.

For each of the queries, the TREC 2010 Web Track provides relevance scores for the documents that can normally be retrieved by this query from the test collection. In the experiments, if a given document retrieved by the underlying search engine cannot be found in the relevance judgments under the corresponding query (different search engines may produce different search results under the same query, and some of the retrieved documents may not be listed in the relevance judgments), this document was considered irrelevant to the query. The relevance judgements were provided on a

six-point scale (from -2 to 3) to show the extent to which the document is relevant. However, for the purpose of this evaluation, it is enough to just have a binary judgment that determines a document to be either *relevant* or *irrelevant* to the intents of the search query. Therefore, the original six-point scale relevance scores were translated into relevant if the score is larger than zero, or irrelevant if the score is less or equal to zero in this evaluation. That is, for this evaluation, any degree of relevance is deemed to be sufficient to consider the document a good document for the query.

In order to conduct the evaluation, a search engine is needed to retrieve the documents from the document collection according to the input query, and provide an initial ranking of the search results. Such a search engine [57] is provided by the creators of the Clue Web09 Dataset, based on the Lemur toolkit [58]. For each search query, this search engine can return up to 1000 search results in a ranked order. The information extracted for each of these search results are the title, snippet, and URL. Also, there is a unique document ID for each result document that can be used to identify the corresponding relevance judgment. In this evaluation, a search for each of the ambiguous test queries was conducted using this search engine. For each of the search results sets, the documents were matched to the relevance scores provided by the TREC 2010 Web Track, and locally cached for use in the evaluations. In this process, a complication arose, which was that some search results contain embedded JavaScript and CSS elements in the snippets. It seems that the search engine has difficulty stripping them out properly. As a result, a manual data cleaning was performed for the documents that were used within the experiments after the search results were stored in the local cache.

Once the test data was prepared, the experiments started with a training phase of

the topic profiles. In the beginning of the training phase, topic profiles were created in the prototype for each of the test queries. The topic profiles were then updated by clicking on the first 20 relevant documents (judged by the provided relevance judgments) that appear after the top-100 documents in the ranked search results list. The assumption here is that these relevant documents that are deep in the results list (after top-100) would not normally be considered by a searcher for the given query (since they are not at the top). However, these documents could appear among the top search results and be viewed if the searcher conducts related searches with similar queries. Therefore, using these documents to populate the topic profiles is intended to mimic a searcher's past search activity for a topic. Note that in the training phase, although only the top-100 search results and the following first 20 relevant documents were used, it was necessary to cache all search results returned by the search engine for each query because training the TF*IDF profiles needs not only the information of the relevant document that is being clicked, but also the information about the surrounding documents of this relevant document on the same page (as described in Section 4.2.3).

After the topic profiles were trained, the experiments entered the evaluation phase. Here, the top-100 documents returned by the search engine for each test query were re-ranked using different methods. The re-ranking results were then compared using the evaluation metrics calculated based on the relevance scores of the documents. Note that none of the documents used in the training phase were used in the evaluation phase, since the evaluation occurred only over the top-100 documents.

Both precision and average precision [6] measured over the top-10 and top-20 documents were used as the evaluation metrics (i.e, P-10, P-20, AP-10 and AP-20).

The formula for calculating precision is as follow:

$$P = \frac{\text{number of relevant documents}}{\text{number of retrieved documents}} \tag{5.1}$$

The formula for calculating average precision is given below:

$$AP = \frac{\sum_{r=1}^{n}(P(r) \times rel(r))}{\text{number of relevant documents}} \tag{5.2}$$

where r is the rank in the list of retrieved documents, n is the number of retrieved documents, $P(r)$ is the precision measured over top r documents in the list, and $rel(r)$ is a function that equals to 1 if the item at rank r is a relevant document, or 0 otherwise.

Precision is simply the ratio of relevant documents to the total documents retrieved (10 and 20 for the experiments). Average precision provides a score that not only takes into account the relevance of the documents, but also their placement within an ordered list. This metric provides a measure of the quality of the ranked search results list, indicating the extent to which the relevant documents are placed in the high positions in the list. Also, average precision is sensitive to small changes in the ranking. A single exchange of ranks between a relevant document and a irrelevant document will change the final average precision score, but will not have an effect on the precision score as long as this exchange is within the scope that the precision is calculated. As such, average precision is a better metric than precision. In the evaluation, average precision was used as the primary metric because of its advantages over precision. However, one of the difficulties with average precision is that it can report a high score even if the precision is low (i.e., when there are a small number of relevant documents, but they are at the top of the list). As such, precision is needed

as a supporting metric to be reported together with average precision so that such special cases can be identified.

Six different re-ranking algorithms were evaluated against each other and in comparison to the original ranked order of the search results in these experiments. The TF approach represents the method employed in the original miSearch system. The TF*IDF approach follows the classical IR approach to improve TF weighting. The TS approach is a result of performing the Luhn-inspired vector re-weighting method proposed in this thesis. As an alternative to replacing the TF vector values with TS values, the fourth approach under investigation scales the TF values by the TS factor, resulting in a TF*TS approach.

For the TS and TF*TS approaches, two tuning parameters (i.e., a and b) are needed in Equation 3.7 for performing the automatic parameter selection (as described in Chapter 3). One may argue that the algorithm is not truly automatic since these two parameters have to be tuned. However, as mentioned previously, the tuning parameters are optional and can be set with default values ($a = 0.1$ and $b = 1.0$). In order to evaluate the tunability of the Luhn-inspired vector re-weighting approach, the untuned re-weighting results will be reported in comparison with the re-weighting results produced by performing tuning on the parameters a and b. These tuned versions of TS and TF*TS represent the fifth and sixth re-ranking algorithms.

5.2 Test Queries

Although 27 queries are provided in TREC 2010 Web Track under the ambiguous category, not all of them can be used as test queries in these experiments. Since 20

relevant documents after top-100 documents are used to populate the personalization vectors in each of the topic profiles, the test collection has to contain at least 20 relevant documents after the top-100 documents for each test query. Also, there must be a sufficient number of relevant documents in the top-100 documents (at least three relevant documents), otherwise it might be difficult to differentiate and compare the performance of the different re-ranking approaches. 17 of the ambiguous queries did not meet these criteria, leaving 10 test queries for use in these experiments (see Table 5.1).

After calculating the AP-10 values of the original ranking for each test query, two queries were removed from the 10 test queries: the topics "ct jobs" and "bart sf". These two topics were excluded from the experiments because they both have very high average precision within the order of the search results provided by the underlying search engine (AP-10 $>$ 0.8). This high average precision means that the search engine produced an outstanding ranking order of the search results, indicating that the query may not be as ambiguous as the creators of the test collection intended. Also, an outstanding original ranking order leaves little room for further improvements made by the personalized re-ranking approaches. Therefore, in the evaluation, these two queries were discarded, resulting in 8 test queries remaining.

Table 5.1: Test queries. Selected from TREC 2010 Web Track "ambiguous" queries.

ID	Query	Description	Relevance Count After Top-100	Original AP-10
52	avp	Find information about events sponsored by AVP, the Association of Volleyball Professionals.	20	0.0
57	*ct jobs*	Find information about jobs in Connecticut.	59	0.989
60	bellevue	Find information about Bellevue, Washington.	75	0.143
63	flushing	Find information about Flushing, a neighborhood in New York City.	48	0.167
77	bobcat	Find dealers that sell or rent Bobcat tractors and construction equipment.	20	0.143
80	keyboard reviews	Find reviews of computer keyboards.	68	0.775
82	joints	Find information about joints in the human body.	33	0.111
86	*bart sf*	Find information about the BART (Bay Area Rapid Transit) system in San Francisco.	20	0.827
96	rice	Find recipes for rice.	45	0.399
97	south africa	Find information about the history, culture, and geography of South Africa.	26	0.381

Certainly it would have been better to have more queries upon which to base the evaluation experiments. However, while the low number of the test queries may reduce the reliability of the statistical analysis of the results, it is expected that a pattern of performance can still be identified in the evaluation, with respect to the different personalized re-ranking approaches under investigation.

5.3 Hypotheses

This evaluation is aimed to explore how the proposed Luhn-inspired vector re-weighting approach improves the performance of the baseline system, and how the different settings affects the behavior of the proposed approach. The benefit of integrating Luhn-inspired vector re-weighting into a Web search personalization system will be measured from the following aspects:

- *Effectiveness*

  Hypothesis 1: *Compared to the original ranking, both TS and TF*TS approaches will produce better ranking orders of the search results.*

  Hypothesis 2: *Compared to the TF (miSearch) ranking, both TS and TF*TS approaches will produce better ranking orders of the search results.*

  Hypothesis 3: *Compared to the TF*IDF ranking, both TS and TF*TS approaches will produce similar quality ranking orders of the search results.*

- *Alternative approaches*

  Hypothesis 4: *TF*TS approach will produce better ranking orders than TS approach.*

- *Tunability*

  Hypothesis 5: *The tuned parameters for automatic parameter selection will produce better ranking orders than the untuned parameters, in both TS and TF*TS approaches.*

5.4 Evaluation Results

5.4.1 Default Tuning Parameters and Tuned Parameters

The experiments for evaluation were conducted in two rounds in order to evaluate the effect of the tuning parameters. The first round of experiments were conducted using default tuning parameters ($a = 0.1$ and $b = 1.0$) for both TS and TF*TS approaches. These default values are educated assumptions for the tuning parameters based on their roles in the formula (as described in Section 3.5.3), and the result of using them represents the performance of the Luhn-inspired vector re-weighting approach in the untuned condition.

A second round of experiments were conducted using the same test data and test queries, but with tuned parameters. In this setting, the two tuning parameters were optimized by PSO (as introduced in Section 3.5.1) based on all test queries, and set to be $a = 0.951$ and $b = 0.882$. That is, one set of tuning parameters was determined based on data from all eight test queries, their search results, and the expert relevance judgements that were provided by the test collection. These parameters are not optimal for any one query, but represent an average optimization over the evaluation set. These parameters were then applied to both TS and TF*TS

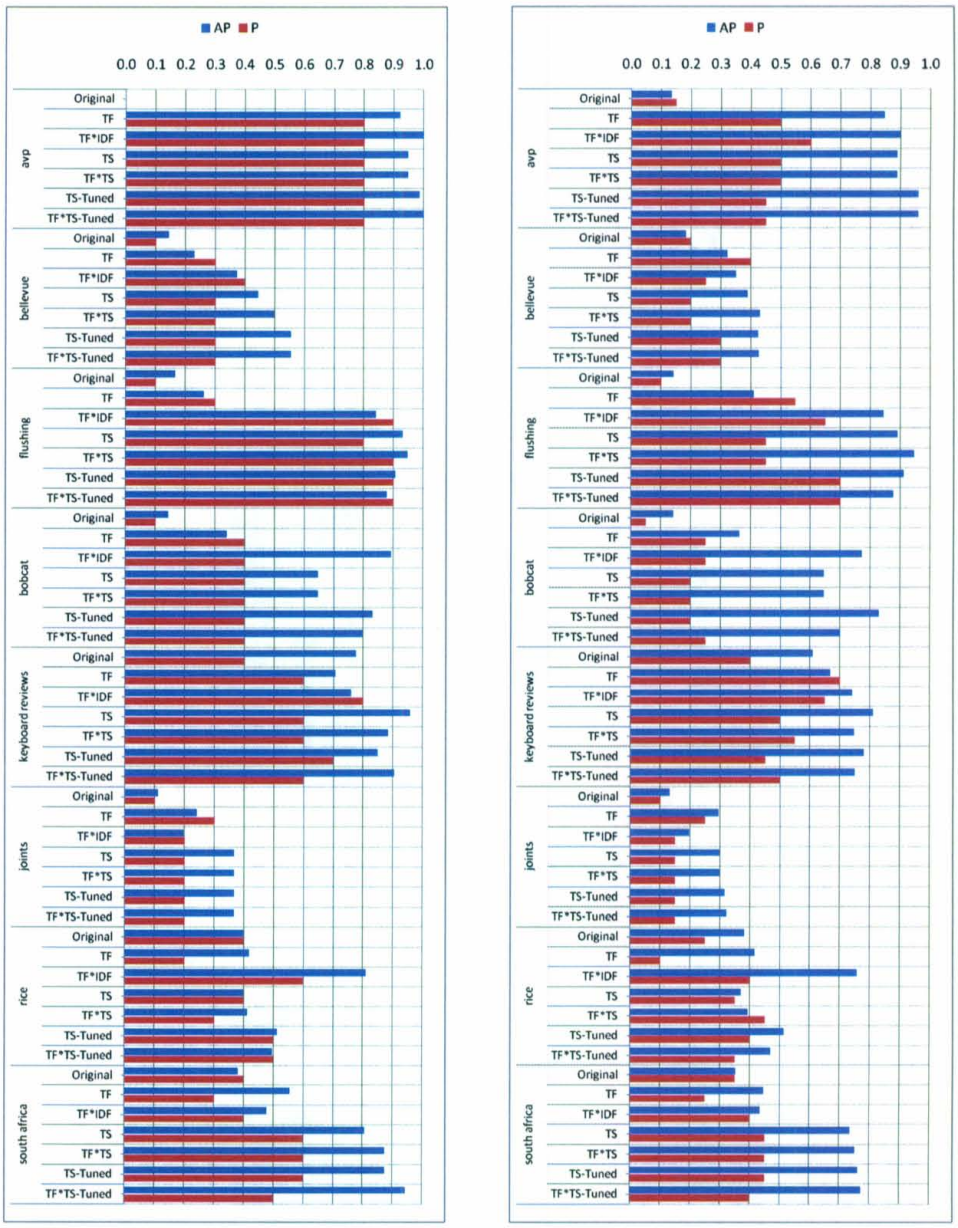
to conduct the re-weighting, and new AP and P values were calculated from the re-ranking results produced by the tuned TS and TF*TS (denoted as TS-Tuned and TF*TS-Tuned) approaches. The purpose of this round of experiments was to measure the potential of the proposed approach once the tuning parameters were properly set.

Since the default setting of the tuning parameters was generated from the definitions of these parameters, this default setting has no bias towards either the TS approach or the TF*TS approach. However, in the tuned case, the tuning was focused on improving the AP metric of the TS approach (i.e., the TS approach and the AP metric were chosen to implement the fitness evaluation in the PSO), so the tuning was inherently biased towards the TS approach and the AP metric. This fact needs to be taken into account when comparing the TS approach to the TF*TS approach in the tuned condition.

5.4.2 Raw Data

The raw data of the evaluation results is presented in this section. For each of the seven approaches that produced ranking lists of the search results (i.e., original ranking, TF, TF*IDF, TS, TF*TS, TS-Tuned, and TF*TS-Tuned), the quality of the ranking was measured in AP and P values over top-10 and top-20 documents. The AP and P values are shown in Figure 5.1.

From Figure 5.1, it is clear that the AP and P values fluctuated widely from one test query to another. For example, the AP and P values are quite high in some topics, such as "avp", "flushing", and "keyboard reviews", but are low in some others, such as "joints" and "bellevue". Because of the standard deviation across these



(a) Top-10

(b) Top-20

Figure 5.1: Average precision (AP) and precision (P) for each test topic.

values, the mean over the set of test queries does not provide an accurate measure of the performance differences between the different approaches studied. Moreover, it is difficult to conduct statistical analysis directly based on these AP and P values because of the high variance. Consequently, these values are only considered as raw data, and further processing is needed to make the data more comparable (which will be discussed in the next section).

The general theme that appears in the raw data is that the TF approach performed better than the original ranking in most cases, the TF*IDF approach performed even better, and the TS and TF*TS approaches performed better than TF*IDF in some cases, but worse in others. In most cases, the TS and TF*TS approaches performed differently and produced different AP and P values. Compared to TS, TF*TS seems to be a better approach because it was outperformed by TS in only one case, which is "keyboard reviews". Also, it is noticeable that the tuned TS and TF*TS approaches performed better than the untuned approaches in many cases. The topics "avp", "bellevue", "bobcat", "rice", and "south africa" are some examples to show the improvements. From these examples, it appears that the tuning of parameters improved the performance of both TS and TF*TS approaches. However, these direct observations from the raw data are not reliable; whether these observed differences were caused by the different approaches or just by random chance remains unknown at this stage. As such, a normalization of the raw data and a statistical analysis of the normalized data are needed to gain more insight into the data, and provide more reliable conclusions of the experimental results.

A couple of interesting special cases can also be observed within Figure 5.1. For the topic "avp", the original ranking order of search results was very poor, but it was

improved dramatically by the TF approach, and TF*IDF, TS, TF*TS, TS-Tuned, and TF*TS-Tuned further improved the ranking order of the search results. In the initial search results, there were no relevant documents in the top-10 list (AP-10 = 0 and P-10 = 0 for the original ranking), but re-ranking the search results (TF*TS-Tuned as the best one) produced an excellent top-10 list which contained 8 relevant documents (P-10 = 0.8), and all of which were ranked at the top of the list (AP-10 = 1.0). Also, for the topic "flushing", note that TF*IDF made significant improvements on the TF ranking, and TS and TF*TS achieved even better results than TF*IDF. However, the improvements degraded somewhat in the tuned case. Since the tuned parameters are the average optimization across all of the test queries, it is reasonable to assume that they may do worse than the untuned versions in some cases. Another similar example can be found in the topic "keyboard reviews".

5.4.3 Average Improvements over TF Approach

The raw AP and P values varied between cases, and thus are not suitable to be used for aggregating the results or conducting statistical analysis to compare the performance between the different approaches. Therefore, it is necessary to normalize these values into a more stable metric for further analysis.

In this normalization, the TF approach was used as the baseline approach, and the AP and P values produced by all other approaches were normalized based on this approach. A new metric called *improvement over the TF approach* was employed, and it was measured by the division of the AP or P values of an approach by the corresponding values produced by the TF approach in an individual test query.

More formally, for each test query, the improvement of a given approach over the TF approach on AP, which was denoted as AP*, was calculated using the following formula:

$$\text{AP*(approach)} = \frac{\text{AP(approach)}}{\text{AP(TF)}} \tag{5.3}$$

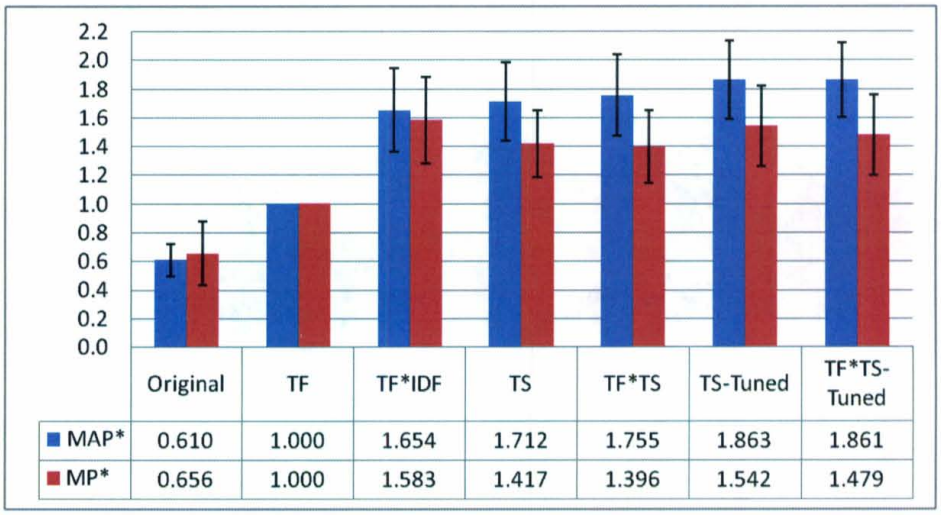
and the improvement over TF on P, which was denoted as P*, was calculated as:

$$\text{P*(approach)} = \frac{\text{P(approach)}}{\text{P(TF)}} \tag{5.4}$$

where approach $\in$ {original, TF, TF*IDF, TS, TF*TS, TS-Tuned, TF*TS-Tuned}. For example, in the query "avp", the AP and P values produced by the TF approach over top 10 documents were AP-10 = 0.921 and P-10=0.8, and TF*TS-Tuned produced AP-10 = 1.0 and P-10 = 0.8. Therefore, the improvement of TF*TS-Tuned over TF on AP-10 was AP*-10 = 1.0/0.921 = 1.086, and the improvement on P-10 was P*-10 = 0.8/0.8 = 1.0.

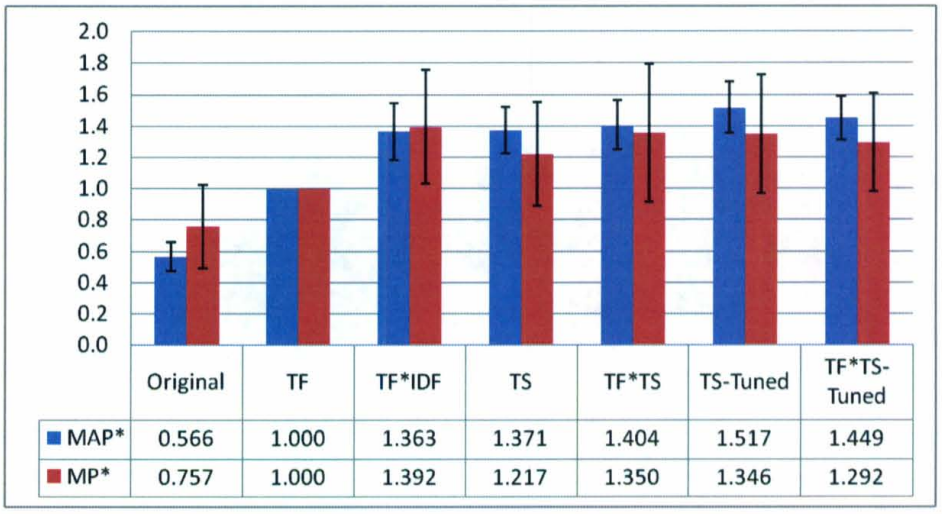
Using this new metric, the arithmetic average of AP* and P* values for each approach were calculated across all eight queries based on both top-10 and top-20 documents. The results are showed in Figure 5.2, where the mean values of AP* and P* are denoted as MAP* and MP*. The error bars in the figure demonstrate the standard errors of the mean values. Note that the TF approach always has a mean value equal to 1.0 and a standard error equal to 0 because it is the baseline approach used in this normalization.

The general trend that emerged from this analysis of average performance change over the TF baseline is that the TF approach improved upon the original ranking order of the search results, and TF*IDF showed even further improvements. Compared to TF*IDF, the (untuned) TS and TF*TS approaches both displayed advantages in



	Original	TF	TF*IDF	TS	TF*TS	TS-Tuned	TF*TS-Tuned
MAP*	0.610	1.000	1.654	1.712	1.755	1.863	1.861
MP*	0.656	1.000	1.583	1.417	1.396	1.542	1.479

(a) Top-10



	Original	TF	TF*IDF	TS	TF*TS	TS-Tuned	TF*TS-Tuned
MAP*	0.566	1.000	1.363	1.371	1.404	1.517	1.449
MP*	0.757	1.000	1.392	1.217	1.350	1.346	1.292

(b) Top-20

Figure 5.2: Average improvements over TF approach regarding to AP (MAP*) and P (MP*). Error bars represent the standard errors of the mean values.

improving MAP* values, but not MP* values. That is, these approaches were able to provide better ranking orders of the relevant documents within the top-10 and top-20 search results, but were unable to draw more relevant documents into these sets from the remaining set of documents. After tuning on the parameters, the TS-Tuned and TF*TS-Tuned approaches performed better and achieved some of the best results. Whether these improvements are statistically significant will be validated in the next section; the discussion here is just based on analysis of the descriptive statistics of the experiments.

MAP* represents the average improvements on the average precision metric that an approach achieved over the TF approach. In other words, this value shows how well an approach performed on promoting the relevant documents within the top-10 and top-20 scopes to the top of the ranked list of search results, in comparison with the TF approach. As shown in Figure 5.2, the original ranking did a poor job on this, probably because of the ambiguous test queries selected for this evaluation. Compared to the original ranking, the TF approach achieved 63.9% improvements on MAP*-10 and 76.7% improvements on MAP*-20, indicating that TF produced much better ranking orders of the relevant documents existing in the top-10 and top-20 search results. By correcting the over-weighting problem of the TF approach, the more complicated approaches (TF*IDF, TS, TF*TS, TS-Tuned, and TF*TS-Tuned) all outperformed the TF approach with considerable increases on MAP*. Among those approaches, TS-Tuned produced the best MAP* values in both top-10 and top-20 scopes, and TF*IDF yielded the worst results on MAP*. However, the differences between those more complicated approaches are rather slight when compared to the differences between them and the original ranking or the TF approach, indicating

that those approaches are at a very similar level with respect to the capability of improving rank orders of relevant documents over the baseline TF approach.

On the other hand, MP* represents the mean value of the improvements made by an approach over the TF approach on the precision metric, which measures how good the approach is in bringing more relevant documents from the rest of the search results set into the top-10 and top-20 scopes (but does not care how these relevant documents are ranked). On the MP* metric, the original ranking was still less effective than the TF approach. Similar to the results on MAP*, all of the more complicated approaches produced better MP* values than the baseline TF approach. However, the Luhn-inspired methods (TS, TF*TS, TS-Tuned, and TF*TS-Tuned) were no longer the best ones in this case, and TF*IDF demonstrated its superiority over other approaches in this group on increasing the number of relevant documents by its highest MP* values among all the approaches in both top-10 and top-20 scopes.

The MAP* and MP* can be used together to evaluate the overall quality of the re-ranking results of the different approaches. Compared to the original ranking and the TF approach, the Luhn-inspired methods (TS, TF*TS, TS-Tuned, and TF*TS-Tuned) all produced better ranking results since the improvements of the MAP* were simultaneously supported by improvements on MP*. That is, not only was the order of the relevant search results improved when viewing the top-10 and top-20 search results, but more relevant documents from the remainder of the set were moved to prominent locations in the search results list. However, when compared to TF*IDF, the improvements on the re-ranking results carried out by the Luhn-inspired methods was not so clear. Although the Luhn-inspired methods were superior in terms of the average precision metric, TF*IDF yield better precision among the top-10 and top-

20 search results. That is, the search results produced by TF*IDF contained more relevant documents than other approaches, but these relevant documents may not be ranked in such a satisfactory order as in the search results list produced by the Luhn-inspired methods.

As discussed in Section 5.4.1, the default tuning parameters are considered unbiased. Therefore, it was expected that TF*TS approach can perform better than TS approach in this untuned scenario. The reason behind this expectation is the fact that TF*TS makes use of the extra information from TF, rather than simply discarding TF information as TS does. While the TF information may sometimes be misleading since the common terms are over weighted, it is still a valuable measure of the importance of the terms. As such, it might be better to integrate TF with TS, rather than just using TS alone. This expectation was supported by the results showed in Figure 5.2, indeed the TF*TS approach produced better MAP* and MP* values than TS approach did (except for MP*-10). However, without the support from statistical analysis, it is not yet sufficient to conclude that TF*TS is a better method than TS.

Compared to the untuned TS and TF*TS approaches, the corresponding TS-Tuned and TF*TS-Tuned approaches produced better MAP* and MP* values on both the top-10 and top-20 scopes, with only one exception on MP*-20, where TF*TS-Tuned produced an average improvement value lower than the untuned TF*TS approach. Therefore, it seemed that the tuning of parameters improved the performance of the Luhn-inspired vector re-weighting. However, further statistical analysis is needed to verify if it is really the case, or this improvement is just a result of the specific features of the different test queries used in these experiments. Another in-

teresting observation is that the TS-Tuned approach yielded better results than the TF*TS-Tuned approach, which might be an effect of the tuning that was biased towards TS approach. However, this biased tuning not only improved the performance of the TS approach, but also increased the average improvement values of the TF*TS approach, indicating that this tuning was proper for both TS and TF*TS approaches.

5.4.4 Statistical Analysis

A statistical analysis was conducted on the average precision (AP*) and precision (P*) improvements over the baseline TF approach, in order to verify statistically significant differences between the approaches that were being evaluated. This statistical analysis was performed using ANOVA tests at a significance level of $\alpha = 0.05$. The reason for choosing ANOVA is that the number of data points in the tests was relatively low (only eight test queries), and ANOVA is known for its robustness with respect to limited data.

In these tests, the raw average precision and precision values (i.e., AP-10, AP-20, P-10, and P-20) produced by the original ranking and each of the re-ranking approaches (TF, TF*IDF, TS, TF*TS, TS-Tuned, and TF*TS-Tuned) across all test queries were normalized through the conversion into the *improvement over the TF approach* metric (i.e., AP*-10, AP*-20, P*-10, and P*-20), and tested in a pair-wise manner between approaches. The results of the statistical analysis are reported in Table 5.2.

Table 5.2: Statistical analysis on average precision and precision improvements over TF approach using pair-wise ANOVA tests. Statistically significant differences ($p < 0.05$) and similarities ($p \geq 0.95$) are marked in bold fonts.

AP*-10	Original	TF	TF*IDF	TS	TF*TS	TS-Tuned	TF*TS-Tuned
Original							
TF	**F(1, 15) = 10.733,p < 0.01**						
TF*IDF	**F(1, 15) = 9.77,p < 0.01**	F(1, 15) = 4.388, p = 0.055					
TS	**F(1, 15) = 12.259,p < 0.01**	**F(1, 15) = 5.97, p = 0.028**	F(1, 15) = 0.019, p = 0.892				
TF*TS	**F(1, 15) = 12.423,p < 0.01**	**F(1, 15) = 6.237, p = 0.026**	F(1, 15) = 0.055, p = 0.818	F(1, 15) = 0.01, p = 0.92			
TS-Tuned	**F(1, 15) = 15.555,p < 0.01**	**F(1, 15) = 8.581, p = 0.011**	F(1, 15) = 0.239, p = 0.633	F(1, 15) = 0.132, p = 0.721	F(1, 15) = 0.065, p = 0.802		
TF*TS-Tuned	**F(1, 15) = 17.16,p < 0.01**	**F(1, 15) = 9.618, p < 0.01**	F(1, 15) = 0.246, p = 0.628	F(1, 15) = 0.135, p = 0.718	F(1, 15) = 0.066, p = 0.801	**F(1, 15) = 0, p = 0.995**	

P*-10	Original	TF	TF*IDF	TS	TF*TS	TS-Tuned	TF*TS-Tuned
Original							
TF	F(1, 15) = 2.086, p = 0.171						
TF*IDF	**F(1, 15) = 5.411, p = 0.036**	F(1, 15) = 3.33, p = 0.089					
TS	**F(1, 15) = 4.853, p = 0.045**	F(1, 15) = 2.778, p = 0.118	F(1, 15) = 0.169, p = 0.688				
TF*TS	F(1, 15) = 4.207, p = 0.059	F(1, 15) = 2.136, p = 0.166	F(1, 15) = 0.2, p = 0.661	**F(1, 15) = 0.003, p = 0.956**			
TS-Tuned	**F(1, 15) = 5.344, p = 0.037**	F(1, 15) = 3.259, p = 0.093	F(1, 15) = 0.009, p = 0.926	F(1, 15) = 0.102, p = 0.754	F(1, 15) = 0.13, p = 0.724		
TF*TS-Tuned	**F(1, 15) = 4.658, p = 0.049**	F(1, 15) = 2.588, p = 0.13	F(1, 15) = 0.057, p = 0.815	F(1, 15) = 0.026, p = 0.875	F(1, 15) = 0.043, p = 0.839	F(1, 15) = 0.022, p = 0.885	

AP*-20	Original	TF	TF*IDF	TS	TF*TS	TS-Tuned	TF*TS-Tuned
Original							
TF	**F(1, 15) = 19.234, p < 0.01**						
TF*IDF	**F(1, 15) = 13.221, p < 0.01**	F(1, 15) = 3.444, p = 0.085					
TS	**F(1, 15) = 18.724, p < 0.01**	**F(1, 15) = 5.554, p = 0.034**	**F(1, 15) = 0.001, p = 0.973**				
TF*TS	**F(1, 15) = 18.256, p < 0.01**	**F(1, 15) = 5.699, p = 0.032**	F(1, 15) = 0.026, p = 0.874	F(1, 15) = 0.02, p = 0.889			
TS-Tuned	**F(1, 15) = 22.418, p < 0.01**	**F(1, 15) = 8.751, p = 0.01**	F(1, 15) = 0.346, p = 0.566	F(1, 15) = 0.382, p = 0.547	F(1, 15) = 0.214, p = 0.651		
TF*TS-Tuned	**F(1, 15) = 24.414, p < 0.01**	**F(1, 15) = 9.106, p < 0.01**	F(1, 15) = 0.123, p = 0.731	F(1, 15) = 0.127, p = 0.726	F(1, 15) = 0.039, p = 0.847	F(1, 15) = 0.088, p = 0.771	

P*-20	Original	TF	TF*IDF	TS	TF*TS	TS-Tuned	TF*TS-Tuned
Original							
TF	F(1, 15) = 0.732, p = 0.407						
TF*IDF	F(1, 15) = 1.733, p = 0.209	F(1, 15) = 1.011, p = 0.332					
TS	F(1, 15) = 1.02, p = 0.33	F(1, 15) = 0.371, p = 0.552	F(1, 15) = 0.11, p = 0.745				
TF*TS	F(1, 15) = 1.164, p = 0.299	F(1, 15) = 0.553, p = 0.469	F(1, 15) = 0.005, p = 0.947	F(1, 15) = 0.051, p = 0.824			
TS-Tuned	F(1, 15) = 1.42, p = 0.253	F(1, 15) = 0.731, p = 0.407	F(1, 15) = 0.007, p = 0.936	F(1, 15) = 0.058, p = 0.814	**F(1, 15) = 0, p = 0.994**		
TF*TS-Tuned	F(1, 15) = 1.481, p = 0.244	F(1, 15) = 0.757, p = 0.399	F(1, 15) = 0.038, p = 0.849	F(1, 15) = 0.024, p = 0.879	F(1, 15) = 0.01, p = 0.921	F(1, 15) = 0.01, p = 0.92	

On both AP*-10 and AP*-20 metrics, statistically significant differences were measured between the original ranking and each of the re-ranking approaches that personalized the search results (i.e., TF, TF*IDF, TS, TF*TS, TS-Tuned, and TF*TS-Tuned). These results indicate that the improvements made by the re-ranking approaches over the original ranking on AP* metric can be attributed to the advantages of personalized re-ranking approaches. Since all re-ranking approaches made use of the extra information within the personalization profiles, these statistically significant results were expected. However, such statistical significance was not found on the P*-10 metric when comparing the TF and the TF*TS approaches to the original order, and also not on the P*-20 metric for any of the re-ranking approaches compared to the original order. For P*-10, what happened was that the original ranking produced some much better P-10 values than the TF approach (in the cases "rice" and "south africa") and TF*TS approach (in the case "rice"), as shown in Figure 5.1. These special cases resulted in high variance in the data, which diminished the statistical significance. Similarly, the large standard errors for MP*-20 (as illustrated in Figure 5.2) yielded by the original ranking and each of the re-ranking approaches (except for the baseline TF approach) indicate that the variance on P*-20 within the approaches under comparison was considerably high. As a result, no statistical significance was found on the P*-20 metric.

As the baseline approach, the TF approach produced better ranking orders of the search results than the original ranking, but was outperformed by the Luhn-inspired methods (i.e., TS, TF*TS, TS-Tuned, and TF*TS-Tuned) with statistical significance found on both AP*-10 and AP*-20 metrics (i.e., on average precision, as the primary evaluation metric). However, no statistical significance was found on the P*-10 and

P*-20 metrics, indicating that the improvements on precision made by the Luhn-inspired methods over the TF approach were marginal and inconsistent. It is a bit surprising to see that the improvements made by TF*IDF over the TF approach were not statistically significant, because TF*IDF showed considerable increase over TF on both MAP* and MP* metrics, as shown in Figure 5.2. However, note that TF*IDF also had larger standard errors for both MAP* and MP* (illustrated as longer error bars in Figure 5.2) than other approaches, indicating that TF*IDF had higher degree of variance in the data, which brought a negative effect on the statistical analysis.

Compared to the TF*IDF approach, no statistically significant difference was detected for the TS, TF*TS, TS-Tuned, and TF*TS-Tuned approaches on any metric. This result suggests that any difference in performance between TF*IDF and the Luhn-inspired methods cannot be attributed to the differences between the approaches. Interestingly, significant similarity (p value in ANOVA test larger or equal to 0.95) between TF*IDF and TS was measured on the AP*-20 metric, showing that those two approaches are essentially equal in improving the average precision of the top-20 search results.

No statistically significant difference was measured between TS and TF*TS, or between TS-Tuned and TF*TS-Tuned. On the contrary, significant similarities were detected for TS and TF*TS on P*-10 metric, and for TS-Tuned and TF*TS-Tuned on AP*-10 metric. Therefore, although TS and TF*TS (and also TS-Tuned and TF*TS-Tuned) produced different AP and P values in almost every single test query (as shown in Figure 5.1) and showed noticeable differences on MAP* and MP* (as shown in Figure 5.2), the differences in performance between these two methods were not consistent.

Although the tuning of parameters improved the performance of both TS and TF*TS approaches on MAP* and MP* metrics (as illustrated in Figure 5.2), this result was not supported by the statistical analysis, as no statistically significant difference was found when comparing TS-Tuned to TS, or TF*TS-Tuned to TF*TS. Therefore, the conclusion is that these improvements are marginal and may be the result of specific features of the test queries and the test collection, rather than the tuning of the shape of the normal distribution curve used for the re-weighting.

5.5 Computational Complexity Analysis

In this section, the proposed Luhn-inspired vector re-weighting approach will be compared to the TF*IDF approach in terms of complexity, in order to analyze the computational costs of these two different approaches.

In the implementation of the Luhn-inspired vector re-weighting approach, the topic profile vector is used exclusively as the source of information. The algorithm traverses through all the terms in the topic profile vector (suppose n terms) three times: once for re-ranking the terms in descending order of frequency, once for finding the mean term, and once for calculating and assigning the TS values. Therefore, the computational cost of the proposed approach is approximately $3n$, where n is the number of terms in the topic profile vector.

On the other hand, in order to generate a topic profile using TF*IDF that contains the same n number of terms, the algorithm needs to search through the document collection for each of these terms to calculate the IDF values. In the implementation of TF*IDF for this research, IDF is estimated using the 50 search results on the

current page as an estimate for the document collection (as described in 4.2.3). As a result, the computational cost of TF*IDF is approximately $50n$.

Since both $3n$ and $50n$ have the same $O(n)$ complexity once the constant before n is removed, they are considered equal in theory. However, the big difference between the two constants makes the two approaches unequal in practice. Moreover, it is possible that larger collections could be employed to calculate the IDF values in other implementations that approximate TF*IDF for Web search personalization (as in [40]). In these cases, thousands, or even millions of documents have to be searched for each term in order to compute the IDF. Even though efficient search strategies could be employed to address this problem, compared to these common practices of using TF*IDF, the proposed Luhn-inspired vector re-weighting approach demonstrates substantial advantages with respect to computational costs.

5.6 Discussion

In this chapter, an experimental evaluation study was reported. This evaluation was conducted with experiments based on test data and queries selected from the TREC 2010 Web Track. These experiments were designed to evaluate the proposed Luhn-inspired vector re-weighting approach in comparison with the baseline approaches (i.e., original, TF, and TF*IDF). In the experiments, four different methods (i.e., TS, TF*TS, TS-Tuned, and TF*TS-Tuned) of the proposed vector re-weighting approach were evaluated.

The TS and TF*TS approaches demonstrated the performance of the proposed vector re-weighting approach in a default setting. In a generalized application, prob-

ably no knowledge about the data, queries, and relevance judgments can be acquired to tune the performance of the proposed approach, so a default setting is required for the approach to perform the re-weighting automatically. The experiments with TS and TF*TS mimicked this generalized scenario with a "best-guess" of the default values from the definition of the tuning parameters, and were intended to evaluate the proposed approach based on the simplest formulation. The results demonstrated that statistically significant improvements over the baseline original ranking and the TF approach can be made by the proposed vector re-weighting approach, even with an unsophisticated estimation of the tuning parameters. However, no significant difference between the TS and TF*TS approaches and the TF*IDF approach was found. This result indicates that while these new approaches are not better than TF*IDF, nor are they worse. In fact, the proposed approach has produced comparable performance to TF*IDF, without the need to access and process the distribution of terms throughout the collection (or a subset of the collection) in order to address the over-weighting problem of the common terms. Instead, this approach calculates a re-weighting based solely on the features of the personalization vector, without the need to access any supplemental information.

For the TS-Tuned and TF*TS-Tuned approaches, knowledge about the test data, queries, and relevance judgments were employed to tune the proposed approach. This scenario was designed to maximize the performance of the proposed approach, and measure its potential for improving the baseline system with proper tuning. The tuning was done over all of the test queries to arrive at a single set of parameters that is essentially the average over each query. The tuned parameters were not optimal for any one query, but intended to be an improvement over the initial best-guess

parameters. The evaluation results showed that the tuning indeed increased the average improvements of the proposed vector re-weighting approach over the TF approach on average precision and precision (measured in MAP* and MP*, as shown in Figure 5.2). However, this improvement was not statistically significant.

The hypotheses proposed in the beginning of this chapter can now be validated via the results from the experiments discussed above:

Hypothesis 1: *Compared to the original ranking, both TS and TF*TS approaches will produce better ranking orders of the search results.* The evaluation results showed that all Luhn-inspired methods (TS, TF*TS, TS-Tuned, and TF*TS-Tuned) outperformed the original ranking. This can be observed in Figure 5.2, where the TS, TF*TS, TS-Tuned, and TF*TS-Tuned approaches yielded much higher MAP* and MP* values compared to the original ranking (nearly three times higher in some cases). These results suggest that the Luhn-inspired methods can bring more relevant documents into the top-10 and top-20 search results, and those relevant documents can be ranked in a better order in positions nearer to the top of the search results list. Moreover, from the results of the statistical analysis, these improvements made by the Luhn-inspired methods over the original order were statistically significant, with only a few exceptions on P*-10 and P*-20. These statistical analysis results indicate that the improvements were a result of using the Luhn-inspired methods, and they are therefore better approaches to order the search results compared to the original ranking. As such, it is concluded that Hypothesis 1 is validated.

Hypothesis 2: *Compared to the TF (miSearch) ranking, both TS and TF*TS approaches will produce better ranking orders of the search results.* In the evaluation, the TF approach has been set as the baseline for normalizing all other approaches

into the new *improvement over the TF approach* metric. According to this new metric, all of the Luhn-inspired methods (TS, TF*TS, TS-Tuned, and TF*TS-Tuned) produced average values greater than 1.0 in all of the four evaluation metrics (MAP*-10, MP*-10, MAP*-20, and MP*-20), indicating that positive improvements on both average precision and precision were made by the Luhn-inspired methods over the TF approach. In the statistical analysis, statistical significance for these improvements was found on the AP*-10 and AP*-20 metrics, but not on the P*-10 and P*-20 metrics. This result indicates that the Luhn-inspired methods are indeed better approaches than TF on ordering the search results and putting the relevant ones to the top of the search results list, but may not be better for bringing new relevant documents into the top-10 or top-20 search results. Since average precision is a better metric than precision on measuring the quality of the ranking order of search results (as discussed in Section 5.1) and has been chosen as the primary evaluation metric, Hypothesis 2 is validated based on the significant improvements on the AP metric made by the Luhn-inspired methods over the TF approach.

Hypothesis 3: *Compared to the TF*IDF ranking, both TS and TF*TS approaches will produce similar quality ranking orders of the search results.* From Figure 5.2, it was observed that the Luhn-inspired methods (TS, TF*TS, TS-Tuned, and TF*TS-Tuned) demonstrated advantages over the TF*IDF approach on the MAP* metric, and TF*IDF achieved superiority over the Luhn-inspired methods on the MP* metric. However, both of the observations were not supported by the statistical analysis. That is, the differences in performance between the Luhn-inspired methods and TF*IDF were minimal and not statistically significant. Moreover, significant similarity was measured between TS and TF*IDF on P*-10, suggesting that these

two approaches were essentially equal in performance with respect to improving precision within the top-10 search results. Therefore, Hypothesis 3 is validated based on the fact that the slight differences between the Luhn-inspired methods and TF*IDF were not statistically significant, and this hypothesis can be further supported by the existence of statistically significant similarity within these approaches.

Hypothesis 4: *TF*TS approach will produce better ranking orders than TS approach.* As explained in Section 5.4.3, it was expected that TF*TS approach can perform better than the TS approach, because of the extra TF information it employs. This expectation was supported by the data illustrated in Figure 5.2, where TF*TS produced better MAP* and MP* values than TS with the default setting of the tuning parameters. However, the statistical analysis demonstrated that there was no significant difference between the performance of TS and TF*TS, nor between TS-Tuned and TF*TS-Tuned. As such, the data regarding Hypothesis 4 is inconclusive.

Hypothesis 5: *The tuned parameters for automatic parameter selection will produce better ranking orders than the untuned parameters, in both TS and TF*TS approaches.* The TS-Tuned and TF*TS-Tuned approaches performed differently from the TS and TF*TS approaches on re-ranking the search results in almost every single test query (as illustrated by the different AP and P values in Figure 5.1). In many cases, the tuned approaches produced better AP and P values than the untuned approaches. Also, at an average level (Figure 5.2), the TS-Tuned outperformed the TS approach, and the TF*TS-Tuned approach outperformed the TF*TS approach, with respect to both the MAP* and MP* metric on top-10 and top-20 search results (MP*-20 was an exception, where TF*TS performed better than TF*TS-Tuned). However,

the statistical analysis results suggest that these differences were not statistically significant. While the observations from Figure 5.1 and Figure 5.2 do show a marginal improvement, due to the lack of statistical significance, Hypothesis 5 is inconclusive.

This evaluation provided answers to the research question regarding the benefits of Luhn-inspired vector re-weighting for Web search personalization and its effects on the ranking of the search results. The experimental results have shown that the proposed Luhn-inspired vector re-weighting approach (TS, TF*TS, TS-Tuned, and TF*TS-Tuned) significantly improved the ranking order of the search results over the baseline original ranking and the TF approach. More importantly, although the results of these experiments did not show a statistically significant improvement over the performance of the well-known TF*IDF approach that is common in information retrieval research, nor did the performance of this Luhn-inspired vector re-weighting approach cause a performance decrease. The key finding in this evaluation is not that this approach provides a performance increase over TF*IDF, but that it achieves performance on par with TF*IDF, without the need to have direct knowledge of the test collection being searched. In fact, the approach can achieve similar results simply by pre-processing a simple TF histogram of the information to find the discriminating terms, and using these as the primary information by which the search results are re-ranked. Therefore, the proposed Luhn-inspired vector re-weighting approach can be easily implemented as a feasible alternative to TF*IDF in Web search personalization to improve the TF personalization models, while avoiding the difficulties that TF*IDF may have in accessing and processing the large amount of information involved in the computation of IDF in the context of Web search.

Chapter 6

Conclusion and Future Work

6.1 Primary Contributions

Web search personalization is effective for assisting users to seek information that meets their personal interests and preferences. In order to learn users' information needs, personalization systems collect and analyze their information from various sources, such as peer recommendations, user questionnaires, desktop and hard drive documents, and past Web search activities. This information is then used to personalize their Web search activities by providing personalized query augmentation or search results re-ranking.

User profiles are used in Web search personalization to model users' interests and preferences. A common approach to model user profiles is to use high dimensional vectors. In these vectors, each dimension represents a term extracted from the information source that is used to generate the personalization vectors; the magnitude along a given dimension is commonly the term frequency (TF), which measures how

often this given term appears in the information source. Normally, these vectors are used to personalize Web search by re-ranking the search results, based on similarity scores calculated between vectors generated from the search result documents and personalization vectors, which represents the users' information needs.

The key benefit of this common vector-based approach is conceptual simplicity and easy implementation. Furthermore, TF vectors can easily be updated with new knowledge by simple vector addition. They can also be compared to one another as well as to individual document vectors produced from the search results using vector distance metrics such as Euclidean distance or Pearson's correlation coefficient.

However, vector-based models that employ TF weights may suffer from an over-weighting problem of the high-frequency terms. High-frequency terms are highly weighted in the TF vectors, but usually they are too common to be helpful for describing the unique characteristics of users' interests and preferences. Moreover, these high-frequency terms are potentially ambiguous in nature, and thus they can easily diminish the effectiveness of the personalization when they are given high weights in the personalization vectors.

Classical information retrieval has suggested TF*IDF as a solution to this problem. In TF*IDF, the high-frequency terms are down-weighted by the inverse document frequency (IDF), which is a measure of how infrequent a given term occurs across all the documents in the collection. In other words, TF*IDF values those terms that appear frequently in the document that is under investigation, but rarely appear in other documents in the collection. In this way, the high-frequency terms can be scaled down by the fact that common terms tend to be used widely in the collection of documents, and thus yield low IDF values.

TF*IDF is a classical measure of term importance, but there are a couple of difficulties with using TF*IDF in Web search personalization. First, the calculation of IDF in the context of Web search is not always feasible because it requires knowledge of the distribution of terms across the entire Web. Moreover, even if it is possible to estimate the IDF by using an existing collection of Web documents or a subset of the Web instead of the entire Web, the overhead of calculating IDF is still considerably high since every term in the personalization vectors has to be searched in the collection to count the number of documents in which it appears. While this information could be cached for future use, the overhead in calculating the document frequency is traded for an overhead in storage and access. Second, it is difficult to incrementally update personalization vectors that are modeled on TF*IDF since the values in such vectors cannot be directly added. However, it is a common requirement in personalization systems that the personalization vectors be updated incrementally, allowing new information on the users' interests to be added to the existing personalization vectors. Third, if the personalization vectors are modeled using TF*IDF, it is also necessary to generate TF*IDF vectors for each document in the search results set, so these document vectors can be compared to the personalization vectors and similarity scores can be calculated. However, the generation of document vectors is costly since it involves the high overhead of calculating IDF for each term in each of the documents in the search results set.

In this thesis, a novel Luhn-inspired vector re-weighting approach is proposed. Similar to TF*IDF, this approach is intended to address the over-weighting problem of common terms within TF vectors, but from a different perspective. This work is inspired by Luhn's model of term importance, and it directly re-weights the terms in

the target TF vector according to the term significance values, which are calculated by placing a normal distribution curve on the top of the term frequency histogram generated from the target vector. The mean value (which decides the location) and the variance (which decides the shape) of the normal distribution curve used in the re-weighting are automatically computed based on the features of the term frequency histogram; the minimum width of the curve and the rate at which the width increases due to the angle of the slope at the mean point can be tuned through two tuning parameters (as described in Section 3.5.3). After the term significance values are assigned to the terms by the normal distribution curve, two different approaches are available for re-weighting terms in the target vector: the TS approach directly replaces the term frequency (TF) values of terms with the term significance (TS) values, and the TF*TS approach re-weights the target vector by scaling down the term frequency by the term significance. These two approaches, along with their tuned versions (i.e., TS-Tuned and TF*TS-Tuned), were evaluated in comparison with the baseline approaches (i.e., the original ranking, the TF approach, and the TF*IDF approach) in this research.

Compared to the TF*IDF approach, the proposed Luhn-inspired vector re-weighting approach has some advantages when applied to Web search personalization. First, this approach requires no supplemental information other than what is in the personalization vector itself. Therefore, this approach is more feasible than TF*IDF in situations where calculating IDF becomes difficult and costly due to the large amount of information that has to be accessed and processed, such as Web search personalization. Second, this approach is based on TF, and it re-weights the personalization vectors without changing their nature as TF vectors. In fact, whenever a

re-weighting is requested, a copy of the target personalization vector is fetched, and the re-weighting will be performed on this copy. Therefore, the incremental updating of the personalization vectors is unaffected, and can be simply done by TF vector addition. Third, this approach has low calculation overhead, as it uses simple formulas to re-weight the TF values in the personalization vectors, and generates simple TF document vectors of search results for comparison. Finally, this approach can be easily applied within systems that have pre-existing TF vectors because all the information required for the re-weighting is contained in the vectors. However, TF*IDF might have difficulties on working on pre-existing TF vectors, since it requires the knowledge of the original collection from which these old TF vectors were generated, which may not be available in a Web context.

The results of performance evaluations show that both the proposed Luhn-inspired vector re-weighting approach and the TF*IDF approach outperform the original order and the simple TF approach to personalization (but the improvement of TF*IDF over the TF approach is not statistical significant). Moreover, the proposed Luhn-inspired vector re-weighting approach is slightly better than the TF*IDF approach in improving average precision (AP) of the search results, but slightly less capable of improving the precision (P) values. In other words, the proposed approach can make better rank orders of the relevant documents and promote them to the top of the search results list, but TF*IDF is more capable of bring new relevant documents from the rest of the search results set into the top-10 and top-20 ranges. However, the statistical analysis demonstrates no significant difference between the proposed approach and TF*IDF (they even show significant similarity on some metrics), providing evidence of the performance equivalence of these approaches.

In conclusion, Luhn-inspired vector re-weighting can be used as a viable alternative to TF*IDF for Web search personalization. This proposed approach has similar performance results with TF*IDF, but requires access to much less information for correcting the over-weighting problem of TF vectors. In this approach, there is no need to conduct costly calculations to generate IDF, and simple formulas are used to scale down the weights of high-frequency terms based solely on information in the TF vector itself. Moreover, this approach does not affect the simple and low-overhead methods for incrementally updating the personalization vectors, since everything within the approach is based on simple TF modeling.

6.2 Future Work

The Luhn-inspired vector re-weighting approach can be further explored and studied in a number of ways. The first is to conduct user evaluations [28] to study the performance of the approach under realistic settings. For example, specific search tasks could be designed and assigned to the users to conduct searches on the system, and the search results could then be re-ranked through different approaches (TF, TF*IDF, TS, and TF*TS). The effectiveness of each of the approaches may be judged by how quickly and how easily the users can find a given number of documents that they consider relevant. Also, it is possible to conduct field trials, in which a group of users can use the system to do real search tasks that they are interested in. In this case, the users are able to experience the differences between approaches and witness the improvement made by the vector re-weighting. By conducting evaluations with real users, new information could be learned on how efficient the approach is in real-

time system operation, how effective the approach is when working with real users' judgments on document relevance, and how robust the approach is when tested with various user-defined queries and user-generated topic profiles.

The characteristics of the two different re-weighting approaches (TS and TF*TS) can be further studied. Initially, it was expected that the TF*TS approach can perform better because it makes use of the TF information (just as TF*IDF uses TF as well), which is discarded in the TS approach. However, the experiments conducted for this thesis have shown that although TS and TF*TS did produce different re-weighting results from the same starting point in most cases, it is difficult to simply tell which one is better because their performance is case dependent and without statistical significance. Therefore, it is worthwhile to conduct further studies and evaluations of these two approaches, in order to identify conditions that can promote the benefits of using the extra TF information in the TF*TS approach. These conditions might be related to the shape of the term frequency histogram used in the re-weighting. That is, TF*TS might work better in a histogram that has a steep shape (and so a steep normal distribution curve), because in this case the weights of high-frequency terms could be dramatically (and might even be overly) reduced and the TF factor can then become a compensation for that.

More ways of tuning the vector re-weighting approach may also be explored. A feature of the approach is that it is tunable, and experimental results have demonstrated some promises on performance gain once the approach is properly tuned (although the improvement was not statistically significant). The tuning in the evaluation was done by employing PSO to maximize the average performance based on the existing relevance judgements. More studies can be conducted to explore other techniques for

tuning the parameters. For example, it may be possible to tune the parameters based on the features of the query (ambiguous or specific) or the topic profile (sparse or robust), and have different tuning in different conditions. By doing so, the approach can be properly tuned whenever it is applied, and achieve its maximum performance. Also, it is possible to tune the parameters based on a much larger and more diverse set of queries (e.g., using Web search logs), to find a stable setting of the tuning parameters that can be used widely.

Alternative formulas to determine the location and the shape of the normal distribution curve that is used for the re-weighting can be considered and evaluated against the formulas devised in this research. These formulas form the basis of the automatic re-weighting algorithm, and any change made on them may directly affect the performance of the proposed Luhn-inspired vector re-weighting approach. Therefore, it is worthwhile to explore other alternatives of these formulas, in order to find the opportunities for further performance increase within the context of the proposed approach.

In this research, when TS and TF*TS personalization vectors are used to re-rank the search results, the documents in the search results set are simply converted into TF vectors, without further re-weighting to convert them into TS or TF*TS vectors. The main reason for this decision is that the information within in the search results is very limited, containing only small pieces of information (URL, title, and snippets) extracted from the source documents. Therefore, the data in the term frequency histograms generated from these search results will be very sparse, making re-weighting on such histograms potentially inaccurate. It would be interesting to explore the opportunities of using the TS or TF*TS approaches to re-weight the

document vectors in addition to the personalization vectors. Such an evaluation could be conducted simultaneously with the study of alternative methods for the placement and shape of the normal distribution curve.

More studies and experiments can also be conducted to identify the conditions under which the proposed approach performs well or performs poorly. No approach is perfect and without any limitation. It is also expected that this approach will have limitations under certain conditions. For example, the experience gained from this research is that this approach needs sufficient information in the target vector to perform the re-weighting, and if the data in the vector is sparse, it may perform inaccurately. However, what if new information is continually added into the personalization vector and eventually the vector becomes very big and noisy? Can this approach perform well under this circumstance? Questions like this are needed to be addressed in the future study, in order to be aware of the strengths and limitations of this approach.

The exploration of other possible applications of the proposed approach is also under consideration. Although implemented within the content-based multiple-profile framework of miSearch, Luhn-inspired vector re-weighting could be implemented within any personalization method that employs a TF-based vector modeling of information, including collaborative-based personalization frameworks. Also, this approach may be helpful to improve TF-based models used in other fields beyond the scope of Web search personalization.

Bibliography

[1] L. A. Adamic and B. A. Huberman. Zipf's law and the Internet. *Glottometrics*, 3:143–150, 2002.

[2] J. Ahn, P. Brusilovisky, D. He, J. Grady, and Q. Li. Personalized Web exploration with task models. In *Proceedings of the International World Wide Web Conference*, pages 1–10, 2008.

[3] J. Alpert and N. Hajaj. We knew the Web was big.... `http://googleblog.blogspot.com/2008/07/we-knew-web-was-big.html`, 2008.

[4] P. G. Anick and S. Tipirneni. The paraphrase search assistant : Terminological feedback for iterative information seeking. In *Proceedings of the International ACM/SIGIR Conference on Research and Development in Information Retrieval*, pages 153–161, 1999.

[5] R. L. Axtell. Zipf distribution of U.S. firm sizes. *Science*, 293(5536):1818–1820, Sept. 2001.

[6] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval: The Concepts and Terminology Behind Search.* Addison-Wesley, Reading, MA, 2nd edition, 2011.

[7] T. Berners-Lee and R. Cailliau. WorldWideWeb: Proposal for a HyperText project. `http://www.w3.org/Proposal.html/`, 1990.

[8] J. Bhogal, A. Macfarlane, and P. Smith. A Review of ontology-based query expansion. *Information Processing and Management*, 43:866–886, 2007.

[9] A. D. Booth. A Law of occurrences for words of low frequency. *Information and Control*, 10(4):386–393, 1967.

[10] A. Broder. A taxonomy of Web search. *ACM SIGIR Forum*, 36(2):3–10, 2002.

[11] R. Butt. *Introduction to Numerical Analysis Using MATLAB.* Infinity Science Press LLC, Hingham, Massachusetts, 2008.

[12] X. Chen and L. Huang. The research of personalized search engine based on users' access interest. In *Proceedings of Asia-Pacific Conference on Computational Intelligence and Industrial Applications*, pages 337–340, 2009.

[13] P.-A. Chirita, C. S. Firan, and W. Nejdl. Summarizing local context to personalize global Web search. *Proceedings of the International Conference on Information and Knowledge Management*, pages 287–296, 2006.

[14] N. Craswell, C. Clarke, and I. Soboroff. TREC 2010 Web track guidelines. `http://plg.uwaterloo.ca/~trecweb/2010.html`, 2010.

[15] H. Cui, J. Wen, J. Nie, and W. Ma. Query expansion by mining user logs. *IEEE Transactions on Knowledge and Data Engineering*, 15(4):829–839, 2003.

[16] A. Dasdan, P. D'Alberto, S. Kolay, and C. Drome. Automatic retrieval of similar content using search engine query interface. In *Proceedings of the ACM Conference on Information and Knowledge Management*, pages 701–710, 2009.

[17] M. de Kunder. The size of the World Wide Web. `http://www.worldwidewebsize.com/`, July 2010.

[18] Z. Dou, R. Song, and J. R. Wen. A large-scale evaluation and analysis of personalized search strategies. In *Proceedings of the International World Wide Web Conference*, pages 581–590, 2007.

[19] H. P. Edmondson and R. E. Wyllys. Automatic abstracting and indexing survey and recommendations. *Communications of the ACM*, 4:226–234, 1961.

[20] Eurekster. Eurekster Swicki Home. `http://www.eurekster.com`, May 2011.

[21] X. Gabaix. Zipf's law for cities: An explanation. *Quarterly Journal of Economics*, 114(3):739–767, 1999.

[22] B. J. Gao and S. Marcos. Rants : A Framework for rank editing and sharing in Web search. In *International Conference on World Wide Web*, pages 1245–1248, 2010.

[23] S. Gauch, M. Speretta, A. Chandramouli, and A. Micarelli. User profiles for personalized information access. In P. Brusilovsky, A. Kobsa, and W. Nejdl, editors, *The Adaptive Web: Methods and Strategies of Web Personalization*, pages 54–89. Springer-Verlag, Berlin Heidelburg New York, 2007.

[24] Google. Google introduces personalized search services; site enhancements emphasize efficiency. `http://www.google.com/press/pressrel/enhancements.html`, 2004.

[25] Google. Google N-Gram Corpus. `http://googleresearch.blogspot.com/2006/08/all-our-n-gram-are-belong-to-you.html`, 2006.

[26] Google. Google SearchWiki. `http://googleblog.blogspot.com/2008/11/searchwiki-make-search-your-own.html`, 2008.

[27] D. Hinkle, W. Wiersma, and S. Jurs. *Applied Statistics for the Behavioural Sciences.* Houghton Mifflin Company, Boston, 1994.

[28] O. Hoeber. User evaluation methods for visual Web search interfaces. In *Proceeding of International Conference on Information Visualisation*, pages 139–145, 2009.

[29] O. Hoeber and J. Gorner. BrowseLine : 2D timeline visualization of Web browsing histories. In *Proceedings of the International Conference on Information Visualization*, pages 156–161, 2009.

[30] O. Hoeber and C. Massie. Automatic topic learning for personalized re-ordering of Web search results. In *Proceedings of the Atlantic Web Intelligence Conference*, pages 105–116, 2009.

[31] B. J. Jansen and A. Spink. How are we searching the World Wide Web? A comparison of nine search engine transaction logs. *Information Processing and Management*, 42(1):248–263, 2006.

[32] K. Keenoy and M. Levene. Personalisation of Web search. *Intelligent Techniques for Web Personalization*, 3169:201–228, 2005.

[33] D. Kelly and J. Teevan. Implicit feedback for inferring user preference : A bibliography. *ACM SIGIR Forum*, 37(2):18–28, 2003.

[34] J. Kennedy and R. Eberhart. Particle Swarm Optimization. In *Proceedings of IEEE International Conference on Neural Networks*, pages 1942–1948, 1995.

[35] Language Technologies Institute of Carnegie Mellon University. The ClueWeb09 Dataset. `http://boston.lti.cs.cmu.edu/Data/clueweb09`, 2009.

[36] H. Liu and O. Hoeber. A Luhn-inspired vector re-weighting approach for improving personalized Web search. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence - Workshops (International Workshop on Web Information Retrieval Support Systems)*, pages 301–305, 2011.

[37] H. Liu and O. Hoeber. Normal distribution re-weighting for personalized web search. In *Proceedings of the Canadian Conference on Artificial Intelligence - Graduate Student Symposium*, pages 281–284, 2011.

[38] R. Losee. Term dependence: A basis for Luhn and Zipf models. *Journal of the American Society for Information Science and Technology*, 52(12):1019–1025, 2001.

[39] H. P. Luhn. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2:159–165, 1958.

[40] N. Matthijs and F. Radlinski. Personalizing Web search using long term browsing history. In *Proceedings of the International Conference on Web Search and Data Mining*, pages 25–34, 2011.

[41] Q. Mei and K. Church. Entropy of search logs: How hard is search? with personalization? with backoff? In *Proceedings of the International Conference on Web Search and Web Data Mining*, pages 45–54, 2008.

[42] A. Micarelli, F. Gasparetti, F. Sciarrone, and S. Gauch. Personalized search on the World Wide Web. In P. Brusilovsky, A. Kobsa, and W. Nejdl, editors, *The Adaptive Web: Methods and Strategies of Web Personalization*, pages 195–230. Springer-Verlag, Berlin Heidelburg New York, 2007.

[43] Microsoft. Microsoft U Rank. `http://research.microsoft.com/en-us/projects/urank/`, May 2011.

[44] A. Moukas and P. Maes. Amalthaea : An evolving multi-agent information filtering and discovery system for the WWW. *Autonomous agents and multi-agent systems*, 1(1):59–88, 1998.

[45] National Institute of Standards and Technology. TREC 2005 Hard Track. `http://trec.nist.gov/data/t14\_hard.html`, 2005.

[46] National Institute of Standards and Technology. TREC 2010 Web Track. `http://trec.nist.gov/data/web10.html`, 2010.

[47] National Institute of Standards and Technology. TREC Home Page. `http://trec.nist.gov/`, April 2011.

[48] National Institute of Standards and Technology. TREC Tracks. http://trec.nist.gov/tracks.html, April 2011.

[49] Netcraft. May 2011 Web Server Survey. http://news.netcraft.com/archives/2011/05/02/may-2011-web-server-survey.html/, May 2011.

[50] Netscape. The Open Directory Project. http://dmoz.org, May 2011.

[51] J. Nielsen. Personalization is overrated. http://www.useit.com/alertbox/981004.html, 1998.

[52] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the Web. Technical Report. Stanford InfoLab, Stanford University, 1999.

[53] M. L. Pao. Automatic text analysis based on transition phenomena of word occurrences. *Journal of the American Society for Information Science*, 29(3):121–124, 1978.

[54] P. Pirolli and S. K. Card. Information foraging. *Psychological Review*, 106(4):643–675, 1999.

[55] J. Pitkow, H. Schutze, T. Cass, R. Cooley, D. Turnbull, A. Edmonds, E. Adar, and T. Breuel. Personalized search. *Communications of the ACM*, 45(9):50–55, 2002.

[56] M. Porter. An Algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[57] T. L. Project. Category B Interactive Search. http://boston.lti.cs.cmu.edu:8085/clueweb09/search/catb/lemur.cgi, April 2011.

[58] T. L. Project. The Lemur Project. `http://www.lemurproject.org/`, April 2011.

[59] S. Robertson, S. Walker, and M. Hancock-Beaulieu. Okapi at TREC-7: automatic ad hoc, filtering, VLC and Interactive track. In *Proceedings of the Seventh Text Retrieval Conference*, pages 253–264, 1998.

[60] G. Salton and C. Buckley. On the use of spreading activation methods in automatic information. In *Proceedings of the International ACM/SIGIR Conference on Research and Development in Information Retrieval*, pages 147–160, 1988.

[61] D. Shen, Z. Chen, Q. Yang, H. Zeng, B. Zhang, Y. Lu, and W. Ma. Web-page classification through summarization. In *Proceedings of the International ACM/SIGIR Conference on Research and Development in Information Retrieval*, pages 242–249, 2004.

[62] A. Sieg, B. Mobasher, and R. Burke. Web search personalization with ontological user profiles. In *Proceedings of the ACM Conference on Information and Knowledge Management*, pages 525–534, 2007.

[63] B. Smyth, E. Balfe, J. Freyne, P. Briggs, M. Coyle, and O. Boydell. Exploiting query repetition and regularity in an adaptive community-based Web search engine. *User Modeling and User-Adapted Interaction*, 14(5):383–423, 2005.

[64] M. Speretta and S. Gauch. Personalized search based on user search histories. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence*, pages 622–628, 2005.

[65] S. Stamou and A. Ntoulas. Search personalization through query and page topical analysis. *User Modeling and User-Adapted Interaction*, 19(1-2):5–33, 2008.

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

[67] J. Teevan. The re: search engine: simultaneous support for finding and re-finding. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, pages 23–32, 2007.

[68] J. Teevan, C. Alvarado, M. S. Ackerman, and D. R. Karger. The perfect search engine is not enough: A study of orienteering behavior in directed search. In *Proceedings of the International Conference on Human Factors in Computing Systems*, pages 415–422, 2004.

[69] J. Teevan, S. T. Dumais, and E. Horvitz. Investigating the value of personalizing Web search. In *Proceedings of the Workshop on New Technologies for Personalized Information Access*, pages 84–92, 2005.

[70] J. Teevan, S. T. Dumais, and E. Horvitz. Personalizing search via automated analysis of interests and activities. In *Proceedings of the International ACM/SIGIR Conference on Research and Development in Information Retrieval*, pages 449–456, 2005.

[71] J. Teevan, R. Jones, and M. Potts. History repeats itself: Repeat queries in Yahoos logs. In *Proceedings of the International ACM/SIGIR Conference on Research and Development in Information Retrieval*, pages 703–704, 2006.

[72] J. Teevan, M. R. Morris, and S. Bush. Discovering and using groups to improve personalized search. In *Proceedings of the International Conference on Web Search and Data Mining*, pages 15–24, 2009.

[73] The World Wide Web Consortium. W3 Servers. `http://www.w3.org/History/19921103-hypertext/hypertext/DataSources/WWW/Servers.html/`, 1992.

[74] C. J. van Rijsbergen. *Information Retrieval.* Butterworths, London, England, 1979.

[75] Q. Wang and S. Jose. Exploring online social activities for adaptive search personalization. In *Proceedings of the International Conference on Information and Knowledge Management*, pages 999–1008, 2010.

[76] R. W. White, I. Ruthven, and J. M. Jose. The use of implicit evidence for relevance feedback in Web retrieval. In *Proceeding of European Colloqium on IR Research*, pages 93–109, 2002.

[77] Y. Zhao, Y. Yao, and N. Zhong. Multilevel Web personalization. In *Proceedings of the International Conference on Web Intelligence*, pages 649–652, 2005.

[78] H. P. Zipf. *Human Behaviour and the Principle of Least Effort.* Addison-Wesley, Oxford, England, 1949.