

Using Community Trained Recommender Models for Enhanced Information Retrieval

Wei Li

B.Sc., M.Sc.

A dissertation submitted in fulfilment of the requirements for the award of

Doctor of Philosophy (Ph.D.)

to the



Dublin City University
School of Computing

Supervisor:

Gareth J. F. Jones

July 2014

Declaration

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of Ph.D. is entirely my own work, that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge breach any law of copyright, and has not been taken from work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

Signed:

(Candidate) ID No.: 58210561

Date:

Acknowledgments

I would like to thank the SFI, Centre for Next Generation Localisation (CNGL) for funding the research of my PhD study, and giving me the support for living and travel to conferences.

I would like to express my sincere gratitude to my supervisor Gareth Jones for his constant support and encouragement. He always guided me with his advice while giving me the freedom and flexibility to pursue my research in various directions. Thanks to you Gareth.

I would like to thank Johannes Leveling for fruitful discussions on my work and his valuable feedback on my thesis, which was a great support while writing it.

I am very grateful to my colleagues in DCU and CNGL, who provided me with help whenever needed. I would like particularly to thank Debasis Ganguly, for his support on my research. And especially thank Maria Eskevich, who is a good friend and always supported me in both my work and life.

Last but definitely not least, I would like to thank the person who this thesis would never have been possible without his support, understanding and patience. I would like to express my sincere gratitude to my soul-mate, my beloved husband Guofeng Li, who always has been my source of power and motivation. Also to my dear parents who always supported and encouraged me in these years. I love you all.

Table of Contents

Acknowledgments	1
Abstract.....	11
1. Introduction	12
1.1 Research Questions	14
1.2 Structure of the Thesis	15
2. Overview of Information Retrieval and Recommender System Techniques and their Combination.....	18
2.1 Information Retrieval.....	19
2.1.1 Basic Processes of Information Retrieval.....	20
2.1.2 How an IR system works.....	21
2.1.3 Retrieval Models.....	24
2.1.4 Query Expansion	30
2.1.5 Personalized Search.....	45
2.2 Recommender Systems	49
2.2.1 Background.....	49
2.2.2 Recommender Algorithms.....	56
2.3 Integrateing IR with RSs.....	66
2.4 Summary	72

3.	An Initial Investigation of an Integrated Information Retrieval-Recommender	
	System Model.....	74
3.1	The Framework of Integrated IR Model	75
3.2	Recommender Component - Topic Category	77
3.3	IR Component - Extended SMART Retrieval System	79
3.3.1	The SMART System	79
3.3.2	The Extended SMART System	80
3.4	Linear Combination	81
3.5	Experimental Results	81
3.5.1	Initial Experiment on INEX 2009	82
3.5.2	Experiment with the TREC-8 Ad hoc Task.....	91
3.6	Summary	106
4.	Investigating the Effectiveness of Alternative Recommender Algorithms and	
	Fusion Methods in the Integrated Information Retrieval Model	108
4.1	Test Collection Building	109
4.1.1	Topic Extraction	110
4.1.2	Collect User Behaviour Data	111
4.1.3	Dwell time extraction	113
4.2	Fusion Methods.....	114
4.3	Experimental Setup	116
4.3.1	IR Component.....	116
4.3.2	Recommender Component	117
4.4	Results	129

4.5	Analysis of the Suitability of Collaborative Filtering Methods for the Integrated	
Model	133
4.5.1	Analysis of Rating-Based CF in the Integrated Model.....	134
4.5.2	Popularity-Focused Rating-Based Collaborative Filtering Method (PFRBCF)	134
4.5.3	Experimental Investigation.....	137
4.6	Summary	138
5.	Exploiting Recommender Techniques in Cluster-Based Link Analysis.....	140
5.1	Introduction	141
5.2	The PageRank Algorithm.....	142
5.2.1	Basic One-Layer Model.....	142
5.2.2	Two-Layer Model.....	144
5.3	Proposed Methods	147
5.3.1	Framework of Integrated Model Exploiting the PageRank Algorithm	148
5.3.2	Using Adjusted Cosine Similarity in Two-Layer Model.....	150
5.3.3	Using Adjusted Cosine Similarity in the Three-Layer Model.....	152
5.3.4	Using Document Deviation in Two-Layer Model.....	155
5.3.5	Using Document Deviation in Three-Layer Model.....	157
5.4	Experimental Investigation	158
5.4.1	Information Retrieval Component.....	158
5.4.2	PageRank Component	158
5.4.3	Experimental Results	164
5.4.4	Cross Validation Experiment.....	167
5.5	Summary	169
6.	Investigation of Query Expansion and Selection of Topic Categories	171

6.1	Introduction	171
6.2	Query Expansion Using Collaborative Filtering Algorithm	173
6.3	Experimental Investigation of Query Expansion	174
6.3.1	Using Pseudo Relevance Feedback for Query Expansion.....	174
6.3.2	Utilizing Recommender Component Output for Query Expansion	176
6.3.3	Log-Based Query Expansion.....	181
6.3.4	Summary for Query Expansion	183
6.4	Investigation of Varying the Number of Topic Categories Used in Recommender- Based Document Ranking.....	184
6.4.1	Analysis of Selecting Multiple Topic Categories for Integrated Model	191
7.	Conclusion and Future Direction	194
7.1	Topic of the thesis	194
7.2	Answer of Research Questions	195
7.3	Contribution of the thesis	198
7.4	Possible Future Directions	199
	Bibliography	201
	Appendix A: Topic Categories and Queries for the Extended FIRE 2011 Test Collection	224
	Appendix B: Instructions for Participants in the FIRE PIR Task.....	228
	Appendix C: Publications	229

List of Figures

Figure 2.1 Basic information retrieval processes (Croft, 1993).....	21
Figure 2.2 An example document.	21
Figure 2.3 The example text after tokenization.	22
Figure 2.4 The example text after stop word removal by SMART list.	22
Figure 2.5 The example text after stemming by Porter's stemmer.....	23
Figure 2.6 An example of merging two postings list.....	24
Figure 2.7 An example of Boolean combinations of sets visualised as Venn diagrams (Hiemstra, 2001).	25
Figure 2.8 Establishing correlations between query terms and documents via query sessions (Cui et al., 2002).	37
Figure 2.9 Three stages of PIR system provide personalized service.....	47
Figure 2.10 Possible relations between means (circles), tendencies (arrows) and prediction (cross) (Cacheda et al., 2011).	62
Figure 2.11 Basis of Slope One schemes: User <i>A</i> 's ratings of two items and User <i>B</i> 's rating of a common item is used to predict User <i>B</i> 's unknown rating (Lemire and Maclachlan, 2005).....	64

Figure 2.12 The HeyStaks system architecture and outline recommendation model (Smyth et al., 2009).....	71
Figure 3.1 Framework of our proposed retrieval model	75
Figure 3.2 The structure of topic model.....	78
Figure 3.3 MAP of different α value when combine IR with RS.....	85
Figure 3.4 Deviation of AP between BL and EIR approaches for 20 original test topics. Calculated as AP (EIR) - AP (BL).....	90
Figure 3.5 an example of the format of TREC-8 topic	95
Figure 3.6 The process of building one topic category for the perfect condition	97
Figure 3.7 The process of building one topic model for the poor condition	100
Figure 4.1 Structure of user logs in N different topic categories	114
Figure 4.2 Example of item-item similarity matrix	124
Figure 4.3 Example of filling each user's profile in the topic category using the cluster mean.....	126
Figure 4.4 The trend of how document frequency and document average rate affect document relevance.	136
Figure 5.1 One-layer link graph (Wan and Yang, 2008)	143
Figure 5.2 Two-layer link graph (Wan and Yang, 2008)	145
Figure 5.3 Workflow of integrated model using PageRank algorithm.....	148
Figure 5.4 Three-layer link graph	153
Figure 5.5 MAP of choosing vary number of documents and additional terms for PRF query expansion.	164

Figure 6.1 Example of similarity in data representation between Collaborative Filtering and Information Retrieval based on the Vector Space Model.	174
Figure 6.2 The MAP of vary parameter values for <i>m</i> using log-based method for query expansion.	182
Figure 6.3 Number of training queries finding the correct topic category when selecting varying <i>X</i> topic categories for each training query.	185
Figure 6.4 MAP of selecting topic categories for each test query to generate document importance rank, and combination with IR output.	189

List of Tables

Table 3.1 Retrieval results for 20 topics with simulated recommender training.....	89
Table 3.2 The results of 7 runs compare to baseline.....	105
Table 4.1 Example of Topic Categories Extracted from Dataset	111
Table 4.2 Example of Topic Categories and User Entered Queries	111
Table 4.3 Example of recorded users search behaviour	112
Table 4.4 Accuracy of selecting the appropriate topic category by using different numbers of documents to generate centroid documents for both current query and topic categories for training set.....	121
Table 4.5 Condition of user tendency and item tendency.....	127
Table 4.6 MAP of 8 different runs using 6 fusion methods.....	130
Table 4.7 MAP of 7 runs, Baseline with content-based CF and one of other 5 CF methods in 6 fusion methods.....	131
Table 4.8 The Relationship between <i>Fi</i> , <i>R</i> and relevance.....	135
Table 4.9 MAP, P@5, P@10 and P@20 value for 7 runs on hybrid approach, merged by CombANZ fusion method.....	137
Table 5.1 Evaluation of all methods	165
Table 6.1 MAP of Pseudo-relevance feedback method for query expansion.....	176

Table 6.2 MAP value for varying the parameter values for M and m using popularity-focused filtering method for query expansion	178
Table 6.3 MAP values for varying the parameter values for M and m using PageRank for query expansion.	179
Table 6.4 Evaluation of using recommender component output for query expansion. ..	180
Table 6.5 Evaluation of using log-based query expansion method, † represents significant improvement over the initial retrieval ranking. * indicates significant improvement over using PRF method for query expansion.	183
Table 6.6 MAP of alternative numbers of topic categories X to generate multiple prediction rankings, and combination with IR output.	188
Table 6.7 MAP of comparison of initial retrieval results with selecting one topic category and selecting two topic categories for each test query and merging using the WLinear+CombANZ method.	190
Table 6.8 The best results we achieved for the proposed integrated IR model in this thesis.	191

Abstract

Research in Information Retrieval (IR) seeks to develop methods which better assist users in finding information which is relevant to their current information needs. Personalization is a significant focus of research for the development of next generation of IR systems. Commercial search engines are exploring methods to incorporate models of the user's interests to facilitate personalization in IR to improve retrieval effectiveness. However, in some situations there may be no opportunity to learn about the interests of a specific user on a certain topic. This is a significant challenge for IR researchers attempting to improve search effectiveness by exploiting user search behaviour.

We propose a solution to this problem based on recommender systems (RSs) in a novel IR model which combines a recommender model with traditional IR methods to improve retrieval results for search tasks, where the IR system has no opportunity to acquire prior information about the user's knowledge of a domain for which they have not previously entered a query. We use search behaviour data from other previous users to build topic category models based on topic interests. When a user enters a query on a topic which is new to this user, but related to a topical search category, the appropriate topic category model is selected and used to predict a ranking which this user may find interesting based on previous search behaviour. The recommender outputs are used in combination with the output of a standard IR system to produce the overall output to the user. In this thesis, the IR and recommender components of this integrated model are investigated.

Chapter 1

Introduction

The ever increasing volume of information available in people's daily lives is creating increasing challenges for document retrieval technologies. One area of growing interest in information retrieval (IR) research is the exploration of methods to enable users to find documents which meet their personal information needs by taking advantage of their previous search history. This is the focus of the area of Personalized Information Retrieval (PIR), which seeks to form a model of each user's search interests using their previous search history, and then to use this model to assist in more reliably retrieving documents of interest to this user in subsequent search activities. Where the user is searching in a topical area of ongoing interest such an approach can prove effective. However, in practice, users may enter queries on new topics which they have not searched on previously. The related field of Recommender Systems (RSs) exploits ratings of items from multiple users to make predictions of items which future users interested in the same topic may find useful.

In recent years, RSs have started to appear in many applications where user feedback is available, for example in online applications such as *YouTube*¹, *Amazon*², and *EBay*³. These systems record the behaviour of users to build interest models of their interests, and use these to predict items which may be of interest to the current user based on feedback from previous ones.

Most existing personalized search engines require personal information from the specific user in order to build a user profile (Mylonas et al., 2008). This data can be collected by asking users to input their personal information preferences, including for example topics of interest or keywords, recording their search queries and clicking and viewing behaviour when browsing retrieved results, or by asking them to rate some items or give other explicit feedback (Yu et al., 2012). In other web search personalization technologies, data is collected without user involvement by exploiting the clustering of retrieved documents in order to create a complete personal user profile based on characterization of their search history (Lu et al., 2007). The approaches introduced above have been found to perform well in the modelled domains (Jansen et al., 1998) (Jeon et al., 2008). However, this approach will not work for new domains where the individual user has not provided personalized information, and it is not realistic to gather such information from users before retrieval operations begin. In this situation it is desirable to make use of any information which is available from previous searchers with similar interests to improve retrieval effectiveness for a user without previous search experiences in the topical domain of their query. To do this, we propose to exploit feedback from previous users search behaviour, either recording explicit feedback of relevance of retrieved items to a query or implicit feedback in terms of the time a user dwells on each item. The feedback information can

¹ <http://www.youtube.com/>

² <http://www.amazon.com/>

³ <http://www.ebay.com/>

then be used to train topic categories for potentially interesting documents for a new searcher who is interested in this topical domain. In this thesis, we introduce an approach to do this by combining recommender technologies with a standard IR method to produce an IR model where user driven models are used to enhance the effectiveness of ranked IR systems.

RSs and IR systems have similar aims: they both attempt to provide users with items which meet their current information needs. For this reason, integrating IR with RSs has become a topic of research interest area in recent years. e.g. (Bellogin and Wang, 2011) (Costa and Roda, 2011), However most of the existing research in this area focuses only on applying collaborative data in IR models or developing a RS which makes use of the concepts and tools in an IR context (Costa and Roda, 2011) (Bellogin and Wang, 2011). This type of combination of IR and RSs is not integrating them in its true sense. Based on these observations, we make an overarching hypothesis which is that *RSs can improve IR system*. We propose an approach to combining recommender technologies with standard IR models to produce an integrated IR model where user driven models are used to enhance the effectiveness of ranked IR systems. Multiple techniques are investigated and examined in this thesis for both the IR component and the recommender component to optimize the performance of the proposed integrated IR model.

1.1 Research Questions

The research questions addressed in this thesis are as follows. For each question, various sub-questions need to be considered.

- Can recommender techniques help IR system to obtain better retrieval results?
- Can user logs be utilized to benefit such a model integrating recommender technique (RT) with an IR model?

- Which recommender algorithms are most suitable in this application?
 - Which existing recommender algorithms benefit the integrated model most?
 - Do existing recommender algorithms provide suitable solutions in this application?
- What is the most effective method to exploit recommender techniques to optimize the results of the integrated IR model?

1.2 Structure of the Thesis

This thesis seeks to answer these research questions and is organized as follows:

Chapter 2 reviews the state-of-the-art in relevant technology for IR and RSs. Both their advantages and shortcomings are examined, and the research opportunity to be explored in this thesis highlighted. We also review existing related work in exploiting user click-through data for improving IR effectiveness and different methods of combining IR and recommender techniques. This description highlights the difference between existing work and the research proposal explored in this thesis.

Chapter 3 describes our initial experiment on our proposed integrated IR model based on simulated user search behaviour extracted from two datasets, INEX 2009 ad hoc task dataset and TREC-8 ad hoc task data collection. The approach of simulating users' search behaviour is introduced in this chapter, with the results obtained for the two datasets and initial conclusions as a result of these initial experiments are described. The challenges of our proposed model are identified and addressed in subsequent chapters.

Chapter 4 contains two parts: it examines the performance of different recommender algorithms (RAs) and investigates the effectiveness of different fusion methods in the integrated

IR model. Existing RAs are described in this chapter and exploited to give recommendations based on the previous users' search logs. Shortcomings of existing RAs in the integrated model are discussed, and a novel adaptive RA is proposed to aid the recommender component in obtaining better recommendation results. It is then demonstrated to improve effectiveness of the combined IR and RS integrated model. Alternative existing fusion methods are explored to perform the combination in the final step of the integrated IR model. Experiments in this chapter are conducted on an extended version of the FIRE 2011 personal information retrieval task test set.

Chapter 5 introduces a novel method of using RAs to aid the IR model. Inspired by the PageRank algorithm for Web search and the existing cluster-based PageRank algorithm, we develop a multi-layer cluster-based PageRank algorithm by using recommender techniques to compute the affinity weight between items based on the search link graph of previous users. We substitute the extracted affinity weight into the PageRank algorithm to compute the importance score for each item. Finally we combine the output PageRank results with IR ranking. The results in this chapter show that this method obtains better performance than the combination of recommender techniques results with IR output, reported in the earlier chapters.

Chapter 6 investigates two main aspects. First, it reports methods for performing query expansion through mining user logs. In this chapter, alternative existing query expansion techniques are investigated. Standard pseudo relevance feedback methods are compared with methods which involve the analysis of user log information into the query expansion process. Second, we investigate the impact of the empirically selected parameters in the integrated IR model. In this chapter, since results in the earlier chapter show sensitivity to selection of the wrong topical category for a query, experiments are conducted which examine selection of different numbers of topic categories for each current query to generate several recommender

rankings using the best performing recommender techniques we identified in the earlier chapters. The multiple prediction rankings are combined with the IR results in the last step of the model. The results show that selecting the two most similar topic categories for a query achieves the best combined results. In brief, this chapter examines the performance of the integrated IR model in different kinds of situations to provide more solid evidence for drawing an overall conclusion of this study.

Chapter 7 concludes the thesis. It highlights the achievements of the thesis and provides directions for possible future work aimed at further improving the effectiveness and efficiency of the integrated IR model which incorporates retrieval relevance information with recommender techniques.

Chapter 2

Overview of Information Retrieval and Recommender System Techniques and their Combination

Both Information Retrieval (IR) and Recommender Systems (RSs) have received considerable interest from the scientific community in recent years, and are widely used to provide users with relevant items in which they are interested. This chapter provides an overview of existing work relevant to the techniques explored in this thesis. The chapter is divided into three parts: IR, RSs and Hybrid approaches which combine IR with RSs techniques together to achieve better results in different specific tasks. For each approach, the basic principles are given first which is followed by a discussion of background information including introduction of relevant notation, and advantages and limitation, finally, relevant existing methods are introduced and described. The methods described in this chapter are employed in the experiments we conduct in this thesis in later chapters. In overview, this chapter aims to give a general introduction and explanation of the approaches, notations and techniques that we use in the thesis.

2.1 Information Retrieval

IR is concerned with the problem of locating documents relevant to a particular user information need from a set of documents. An IR process begins with a user entering a search query which seeks to explain the information need to the retrieval system. The IR system then computes a numeric matching score based on how well each object in the dataset matches the query. The objects are then returned ranked according to this value. In this thesis, we are focus on ad hoc retrieval of text documents.

The challenge we are seeking to address in our work is as follows. In general, not all documents returned by a search engine are relevant to a user's information need. Various techniques have been proposed to address this problem, two of the most popular of which are: query expansion and personalized IR. Query expansion is used to address the query-document mismatch problem which is particularly significant for short queries, often leading between the query and relevant documents. This arises since short queries will often fail to match with sufficient of the terms in relevant documents to enable them to be retrieved at high rank or they may not be retrieved at all. To solve this vocabulary mismatching problem, query expansion techniques can be used to help users form more meaningful or effective queries or expand queries entirely automatically (Efthimiadis, 1996) (Bhogal et al., 2007). Query expansion assists users in formulating a better query by adding additional keywords to the initial search query in order to encapsulate their interests to focus the search output accordingly.

Personalized IR (Agichtein et al., 2006) (Brusilovsky and Tasso, 2004) (Gauch et al., 2007) (Sugiyama et al., 2004) aims at improving the retrieval process by taking into account the particular interest of individual user. A standard IR system usually provides the same document ranked list to different users, even though users may require different search results based on

their different backgrounds or interests in the topic. Personalized IR addresses this problem by capturing personalized context, such as the user's interests, user's profile, user's behaviour, and applying this context information to the search activities for each user. For example, Web pages are personalized based on the characteristics (interests, social category, context,...) of an individual. Personalization implies changes for individual users based on the implicit data they provide, such as items they have purchased before or pages they have viewed previously.

Besides these two, another popular and widely used technique is to exploit web link graphs, for example using the PageRank algorithm (Brin and Larry, 1998) (Larry et al, 1998). The PageRank algorithm was developed by the founders of the Google web search engine, and assigns a numerical weighting to each element of a hyperlinked set of documents, such as those on the World Wide Web (WWW), with the purpose of measuring the relative importance of this element within the set.

The following subsections describe the basic processes of an IR system and how the overall IR system works. After this, since query expansion, personalization and link analysis are all important to our investigations, we introduce each of these in turn, highlighting the principles, underlying the methods and reviewing key work in each area.

2.1.1 Basic Processes of Information Retrieval

There are three basic processes that an IR system has to support: representation of the content of the documents, representation of the user's information need, and comparison of the two representations. The process is shown in Figure 2.1 (Croft, 1993).

The process of representing the information need is often referred to as the query formulation process, the resulting formal representation is the query, and representing the document is usually called the indexing process.

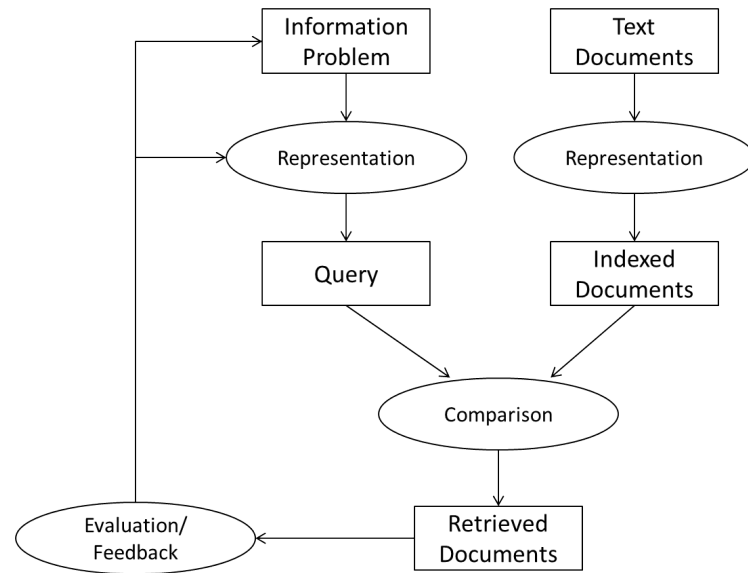


Figure 2.1 Basic information retrieval processes (Croft, 1993).

2.1.2 How an IR system works

Usually, the standard process of an IR system works according to the following steps: first, a tokenisation process is applied, then stop words are removed, and after which the remaining words are stemmed, this may be followed by extraction of phrases. Document are then indexed, following which documents can be retrieved in response to a given query. Figure 2.2 shows a simple text example.

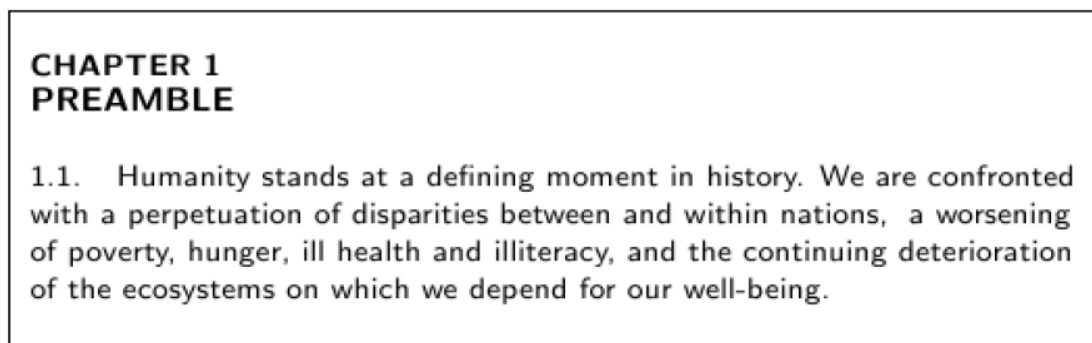


Figure 2.2 An example document.

2.1.2.1 Tokenization

As a first step in the processing of a document or a query, the tokens have to be determined. One of the most simple approaches to tokenisation defines word and inter-word symbols. In the example of Figure 2.3, all characters that are non-letters and non-digits are considered to be inter-word symbols. The inter-word symbols are ignored during this phase, and the remaining sequences of word symbols are the indexing tokens.

```
chapter 1 preamble 1 1 humanity stands at a defining moment in
history we are confronted with a perpetuation of disparities
between and within nations a worsening of poverty hunger ill
health and illiteracy and the continuing deterioration of the
ecosystems on which we depend for our well being
```

Figure 2.3 The example text after tokenization.

2.1.2.2 Stop word removal

```
chapter 1 preamble 1 1 humanity stands defining moment
history confronted perpetuation disparities nations
worsening poverty hunger ill health illiteracy continuing
deterioration ecosystems depend well being
```

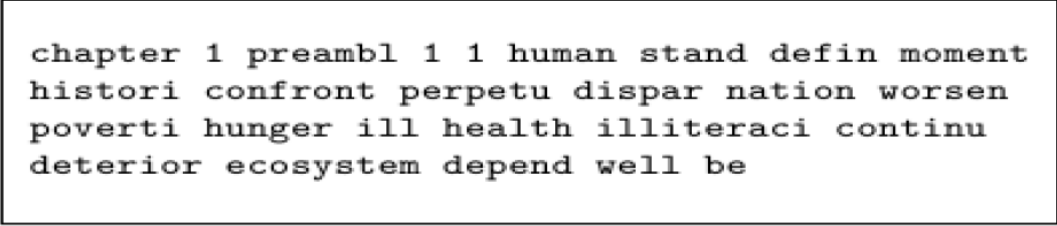
Figure 2.4 The example text after stop word removal by SMART list.

Stop words are words which may carry little meaning from a frequency point of view or alternatively from a linguistic point of view. These words are removed from the documents and the query. Removing stop words for linguistic reasons can be done by using a stop list that enumerates all words with little meaning, for instance “the”, “it” and “a”. Stopword lists are used in many IR systems, but the length of the various stop lists may vary considerably. For instance,

the SMART stop list contains 571 words⁴, whereas the Okapi system uses a moderate stop list of about 220 words (Robertson and Walker, 2000).

2.1.2.3 Stemming

In IR, stemming is the process of reducing inflected words to their stem, base or root form. The stemmers most commonly used in IR are those by Lovin (Lovin, 1993) and Porter (Porter, 1980). A stemmer can produce undesirable effects, for example, it may conflate two words with very different meanings to the same stem. For example “operate”, “operating” and “operations” are all stemmed to “oper”. As a result, a query “operating systems” can fetch documents related to “operations research”. Figure 2.5 shows the result of Porter’s stemmer on the sample text.



```
chapter 1 preamb1 1 1 human stand defin moment  
histori confront perpetu dispar nation worsen  
poverti hunger ill health illiteraci continu  
deterior ecosystem depend well be
```

Figure 2.5 The example text after stemming by Porter’s stemmer.

2.1.2.4 Phrase extraction

During indexing and automatic query reformulation, multiple words may be treated as a single processing token. The meaning of a phrase may be quite different from what the words suggest independently. Maintaining the position information of terms is a generalized approach to dealing with word n-grams, where a document is retrieved from the index if the positional information of the query terms conforms with those in the document. For example the query “to

⁴ Smart, “<ftp://ftp.cs.cornell.edu/pub/smart>”.

be or not to be” is less likely to match to Shakespeare’s “Hamlet” without the positional information.

2.1.2.5 Index file structure

Within a document collection, each document has a unique number known as the document identifier. A weighted list of documents is constructed for every term in the collection, where the weight assigned to a document may be the number of occurrences of that term in the individual document. The terms, which act as keys to their corresponding lists, are kept sorted and are typically kept in memory, whereas the associated lists (commonly referred to as postings) are kept sorted by the list members and weights and are typically stored in secondary storage. The postings of each query term are merged to give the final set of documents. Figure 2.6 shows merging of two postings.

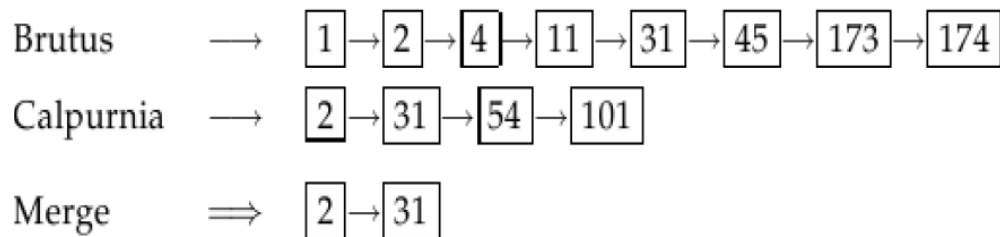


Figure 2.6 An example of merging two postings list.

2.1.3 Retrieval Models

There are two good reasons for having models of IR. The first is that models guide research and provide the means for academic discussion. The second reason is that models can serve as a blueprint to implement an actual retrieval system (Hiemstra, 2001). The main purpose of all retrieval models is to compute the similarity between documents and queries. The following subsections describe in outline four commons IR models in order in which they were developed.

2.1.3.1 Boolean model

The Boolean model was the first model of IR and probably also the most criticised model. This model can be explained by thinking of a query term as a unambiguous definition of a set of documents. Boole defined three basic operators, the logical product called AND, the logical sum called OR and the logical difference called NOT. The Boolean model allows users to specify their information need using a complex combination of Boolean AND, OR and NOT operators. The results set can be visualised in Venn diagrams. Figure 2.7 shows an example of this method for a simple query.

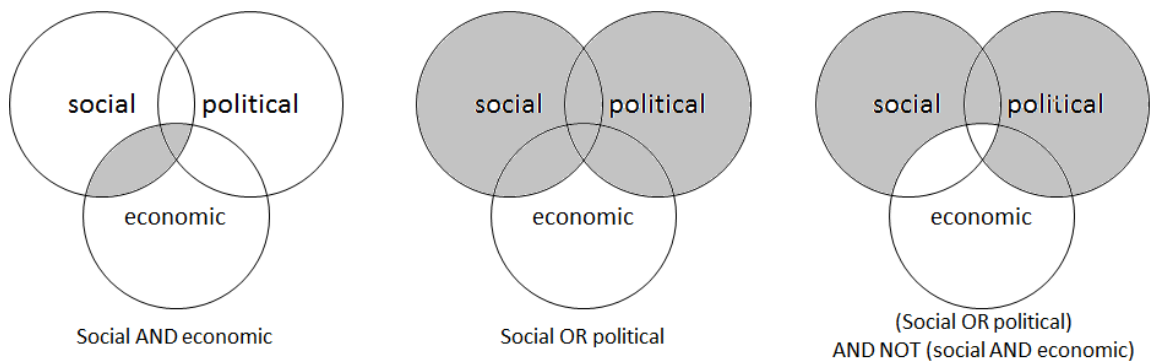


Figure 2.7 An example of Boolean combinations of sets visualised as Venn diagrams (Hiemstra, 2001).

An advantage of the Boolean model is that it gives (expert) users a sense of control over the system. Its main disadvantage is that it does not provide a ranking of retrieved documents. The model either retrieves a document or not, which may lead to users becoming frustrated as they seek relevant information with no suggestion of which document they should look at next.

2.1.3.2 Vector space model

Luhn was the first to suggest a statistical approach to searching information (Luhn, 1957). He proposed that in order to search a document collection, the user should first prepare a document that is similar to the documents needed. The degree of similarity between the representation of

the prepared document and the representations of the documents in the collection would be used to rank the search results. Luhn formulated his similarity criterion as follows: *The more two representations agree in given elements and their distribution, the higher would be the probability of their representing similar information.*

Following Luhn's similarity criterion, a promising first step is to count the number of elements that the query and the index representation of a document share. If a document's index representation is a vector $\vec{d} = (d_1, d_2, \dots, d_m)$ of which each component $d_k (1 \leq k \leq m)$ is associated with an index term, and if the query is a similar vector $\vec{q} = (q_1, q_2, \dots, q_m)$ of which the components are associated with the same terms, then a straightforward similarity measure is the vector inner product (or dot-product) defined as shown in Equation (2-1).

$$score(\vec{d}, \vec{q}) = \sum_{k=1}^m d_k \cdot q_k \quad (2-1)$$

If the vector has binary components, i.e. the value of the components is 1 if the term occurs in the document or query and 0 if not, then the vector product measures the number of shared terms. A more general representation would use natural numbers or real numbers for the components of the vectors \vec{d} and \vec{q} .

Gerard Salton and his colleagues suggested a model based on Luhn's similarity criterion that has a stronger theoretical motivation (Salton and McGill, 1983). They considered the index representations and the query as vectors embedded in a high dimensional Euclidean space, where each term is assigned a separate dimension. The similarity measure is usually the cosine of the angle that separates the two vectors \vec{d} and \vec{q} . The cosine of an angle is 0 if the vectors are orthogonal in the multidimensional space and 1 if the angle is 0 degrees. The cosine formula is given by Equation (2-2).

$$score(\vec{d}, \vec{q}) = \frac{\sum_{k=1}^m d_k \cdot q_k}{\sqrt{\sum_{k=1}^m (d_k)^2 \cdot \sum_{k=1}^m (q_k)^2}} \quad (2-2)$$

The metaphor of angles between vectors in a multidimensional space makes it easy to explain the implications of the model to non-experts.

The main disadvantage of the vector space model is that it does not in any way define what the values of the vector components should be. The problem of assigning appropriate values to the vector components is known as term weighting. Early experiments by Salton (1971) and Salton and Yang (1973) showed that term weighting is not a trivial problem. They suggested so-called *tf.idf* weights, a combination of term frequency *tf*, which is the number of occurrences of a term in a document, and *idf*, the inverse document frequency, which is a value inversely related to the document frequency *df*, which is the number documents that contain the term. Many modern weighting algorithms are versions of family of *tf.idf* weighting algorithms. Salton's original weights perform relatively poorly, and are defined as shown in Equation (2-3).

$$d_k = tf(k, d) \cdot \log \frac{N}{df(k)} \quad (2-3)$$

where $tf(k, d)$ is the number of occurrences of the term k in the document d , $df(k)$ is the number of documents containing k , and N is the total number of documents in the collection.

2.1.3.3 Probabilistic model

Maron and Kuhns (1960) introduced ranking by the probability of relevance. Robertson turned this idea into the probability ranking principle (PRP), which he attributed to Cooper (Robertson, 1977): The PRP is defined by Robertson as follows. *If a reference retrieval system's response to each request is a ranking of the documents in the collections in order of decreasing probability of usefulness to the user who submitted the request, where the probabilities are estimated as accurately as possible on the basis of whatever data has been made available to the system for*

this purpose, then the overall effectiveness of the system to its users will be the best that is obtainable on the basis of that data.

Robertson and Spärck Jones based their probabilistic model of IR on this line of reasoning (Robertson and Spärck Jones, 1976). They suggested ranking documents by $P(R|D)$, that is the probability of relevance R given the document's content description D , where D is a vector of binary components, each component typically representing terms. In the probabilistic retrieval model the probability $P(R|D)$ has to be interpreted as follows: there might be several, for example 10, documents that are represented by the same D . If 9 of them are relevant, then $P(R|D) = 0.9$. Bayes' rule is used on the probability odds $P(R|D)/P(\bar{R}|D)$, where \bar{R} denotes irrelevance. The odds allow us to ignore $P(D)$ in the computation, while still providing a ranking by the probability of relevance. Additionally, this model assumes independence between terms given relevance, as shown in Equation (2-4).

$$\frac{P(R|D)}{P(\bar{R}|D)} = \frac{P(D|R)P(R)}{P(D|\bar{R})P(\bar{R})} = \frac{\prod_K P(D_k|R)P(R)}{\prod_K P(D_k|\bar{R})P(\bar{R})} \quad (2-4)$$

where D_k denotes the k^{th} component (term) in the document vector. A more convenient implementation of probabilistic retrieval uses the following three order preserving transformations. First, the documents are ranked by sums of logarithmic odds, instead of the odds themselves. Second, the a priori odds of relevance $P(R)/P(\bar{R})$ is ignored. Third, the term $\sum_k \log (p(D_k = 0|R)/P(D_k = 0|\bar{R}))$. This way, the sum over all terms only includes non-zero values for terms that are present in the document, as shown in Equation (2-5)..

$$matching\ score(D) = \sum_{k \in matching\ terms} \log \frac{P(D_k = 1|R)P(D_k = 0|\bar{R})}{P(D_k = 1|\bar{R})P(D_k = 0|R)} \quad (2-5)$$

where $matching\ score(D)$ denotes the relevance score of document D to the query.

2.1.3.4 Language models

A statistical language model assigns a probability to a sequence of m words $P(w_1, w_2, \dots, w_m)$ by means of a probability distribution. Language models (LM) were applied to IR by a number of researchers in the late 1990's (Ponte and Croft, 1998) (Hiemstra and Kraaij, 1998) (Miller et al., 1999). The principle of LM originates from probabilistic models of language generation developed for automatic speech recognition systems in the early 1980's. For IR, language models are built for each document. They take the same starting point as the probabilistic indexing model by Maron and Kuhns which is, given a relevant document D , a user will formulate a query by using a term w with some probability $P(w|D)$. The probability is defined by the text of the documents: If a certain document consists of 100 words, and of those the word "good" occurs twice, then the probability of "good" given that the document is relevant is simply defined as 0.02. For queries with multiple words, it is assumed that query words are generated independent of each other, i.e., the conditional probabilities of the terms T_1, T_2, \dots, T_n given the document are multiplied as shown in Equation (2-6).

$$P(T_1, T_2, \dots, T_n|D) = \prod_{i=1}^n P(T_i|D) \quad (2-6)$$

In Equation (2-6), the probability of a single word T_i which is not contained in the document D , is zero. As a result the joint probability of generating n terms becomes zero. To address this problem, smoothing is used to prevent the presence of query words with zero probabilities. There are many approaches that have been proposed for smoothing, most pioneered for automatic speech recognition (Chen and Goodman, 1996). Another approach to smoothing that is often used for IR is the so-called Dirichlet smoothing, which is defined as shown in Equation (2-7) (Zhai and Lafferty, 2004).

$$P(T_1 = t_1, \dots, T_n = t_n | D) = \prod_{i=1}^n \frac{tf(t_i, d) + \mu P(T_i = t_i)}{(\sum_t tf(t, d)) + \mu} \quad (2-7)$$

where μ is a non-negative real number. Dirichlet smoothing accounts for the fact that documents are too small to reliably estimate a language model. Smoothing using Equation (2-7) has a relatively big effect on small documents, but relatively small effect on bigger documents. The probability of a query given a document and the probability of a document given the query are related by Bayes' rule as shown in Equation (2-8).

$$P(D | T_1, T_2, \dots, T_n) = \frac{P(T_1, T_2, \dots, T_n | D)P(D)}{P(T_1, T_2, \dots, T_n)} \quad (2-8)$$

Since $P(T_1, T_2, \dots, T_n)$ does not depend on the document, ranking the documents by the numerator of the right-hand side of Equation (2-8) will rank them by the probability given the query.

2.1.4 Query Expansion

As mentioned in Section 2.1, query expansion is used to address the short query problem which can lead to a query mismatch with relevant documents. Query expansion techniques have been examined and shown to perform very well by many researchers, (Rocchio and Salton, 1971) (Salton and Buckley, 1990) (Cui et al., 2002) (Lu et al., 2009). There are two key aspects of query expansion techniques: i) the source from which expansion terms are selected; ii) the method of how to select expansion terms (Cui et al., 2002).

2.1.4.1 Background

One of the key issues for successful IR is the matching of terms in a user search request against the contents of relevant documents. Usually, a user's search request brings two challenges for search engines: i) users typically describe their information needs using only a few keywords, as a result, it is a significant challenge for an IR system to match the query with the contents of

relevant documents; ii) the words that users use to describe their information needs are often different from the terms which describe the same concept in the relevant documents. As a consequence of these problems, in many cases the documents returned by search engines are not relevant to the user's information need. This raises a fundamental problem of term mismatch in IR. Query expansion techniques are used to help address this mismatch problem by adding a number of meaningful terms to an initial query to better describe the information need in order to produce a better set of search results (Efthimiadis, 1996) (Bhogal et al., 2007). This process of adding terms can either be manual, interactive or automatic. Manual query expansion relies on a searcher's expertise and knowledge to identify additional terms to add to their queries. Interactive query expansion identifies and presents to the searcher an ordered list of potential query expansion terms, allowing them to explicitly choose which ones to add to their query. Automatic query expansion calculates and assigns weights to a set of candidate terms. Those with the highest weights are added to the initial query without consultation with the user. Since different weighting functions produce different results, retrieval performance depends on how the weights have been calculated for selection of expansion terms.

Since both manual and interactive query expansion methods require user involvement, which is not realistic in some cases because users only want the results, rather than to help the search engine to find relevant results, we will focus more on automatic query expansion in this thesis. Current automatic query expansion techniques can generally be categorized into global analysis and local analysis (Manning et al., 2008).

A query expansion method based on global analysis usually builds a thesaurus to assist users reformulating their queries. A thesaurus can be automatically established by analysing relationships among documents and statistics of term co-occurrences in the documents (Cui et al., 2002). From a thesaurus constructed in this way, one will be able to obtain synonyms or

related terms given a user query. Thus, these related terms can be used for supplementing the user's original query. Global analysis methods are techniques for expanding or reformulating query terms independent of the query and results returned for it, so that changes in the query wording will cause the new query to match other semantically similar terms. Only individual query terms are considered for expansion. Global query expansion methods include:

- Query expansion/reformulation query with a thesaurus or WordNet (Miller et al., 1990).
- Query expansion via automatic thesaurus generation.

Local analysis extracts expansion terms from a subset of the initial retrieval results. This subset may be determined directly by the user according to relevance judgments or automatically retrieved by the system, e.g. by assuming that the top-ranked documents are relevant. Terms selected from these documents are added to or replace the initial query, or their weights may be increased to enhance their contribution to the ranking of retrieved documents in a subsequent retrieval run. The basic local methods are:

- Relevance Feedback
- Pseudo Relevance Feedback
- Indirect Relevance Feedback

The following sections describe these global and local methods in more detail.

Global Methods

Query expansion via global analysis uses some form of thesaurus or knowledge resource (Manning et al., 2008). Each term t in a query can be expanded with synonyms and related terms from the thesaurus. Each term in the expansion can be associated with a weight. Since they are selected automatically, expansion terms selected from a thesaurus or knowledge resource can be

given less weight than the original query terms. Methods for building a thesaurus for query expansion include:

- Use of a controlled vocabulary that is maintained by human editors. These thesauruses contain canonical terms for each concept. The subject heading of traditional library subject indexes, such as the Library of Congress Subject Headings (LCSH)⁵, or the Dewey Decimal system are examples of a controlled vocabulary (Scott, 1998). Use of a controlled vocabulary is quite common for well-resourced domains. A well-known example is the Unified Medical Language System (UMLS) used with MedLine for querying the biomedical research literature.
- Manual thesauruses are constructed by human editors and contain a set of synonymous names for concepts, without designating a canonical term. The UMLS⁶ metathesaurus is an example of such thesaurus. Statistics Canada maintains a thesaurus of preferred terms, synonyms, broader terms, and narrower terms for matters on which the government collects statistics, such as goods and services (Manning et al., 2008).
- An automatically derived thesaurus is developed with manual processing using co-occurring terms or grammatically related terms. Terms which co-occur quite frequently in a corpus are more likely to be related. The other approach is to analyse the corpus for grammatical dependencies. For example, entities that grow, walk, or move are more likely to be living organisms or more specifically, to be humans.

⁵ The LCSH comprise a thesaurus of subject headings, maintain by the United States Library of Congress, for use in bibliographic records. <http://id.loc.gov/authorities/subjects.html>

⁶ The UMLS is a compendium of many controlled vocabularies in the biomedical science (created 1986). It provides a mapping structure among these vocabularies and thus allows one to translate among the various terminology systems. <http://www.nlm.nih.gov/research/umls/>

- Query reformulations based on mining of users' logs. Large amounts of user interaction information are available to web search engines. This information is stored in query logs and can be used to improve the user satisfaction of later users by selecting similar interest user logs by comparing the similarity between later users and existing users logs information.

A large amount of work has explored query expansion using thesauri or knowledge resources. In the following section we review some of this work.

Manually Developed Lexical Resources

Manually developed lexical resources such as the WordNet (Miller et al., 1990) (Voorhees, 1994), UMLS metathesaurus (Olivier, 2004) (Eichmann and Ruiz, 1998) (Lu and Mu, 2009), *etc.* are commonly used for query expansion (Aronson and Rindflesch, 1997). In (Voorhees, 1994), WordNet was used for query expansion. Synonym terms were added to the query. It was observed that this approach makes little difference in retrieval effectiveness if the original query is well formed. However, queries which are not well formed can be improved significantly.

Automatically Derived Thesaurus

There are many approaches which use corpus or lexical resources to automatically develop a thesaurus. Most of these methods are used in domain specific search engines or applications. Qiu and Frei (1993) constructed a term-vs-term similarity matrix based on how the terms of a collection are indexed. A probabilistic method was used to estimate the probability of a term similar to a given query. Even when adding hundreds of terms into queries, this approach showed that the similarity thesaurus can improve performance significantly. Adding so many terms may not, however, be efficient for large IR system. Jing and Croft (1994) used a phrase finder that automatically constructs a thesaurus using text analysis and text feature recognition. Phrase finder considered co-occurrence between phrases and terms as associations. They

conducted this method on thesauri built from the NPL and TIPSTER collections, and reported 30% increase in performance and concluded that it is possible and feasible to construct useful collection-dependent association thesauri automatically without relevance judgements.

Query Expansion Based on Query Log Mining

With the increase in usage of Web search engines, it has become easy for companies operating these systems to collect and use very large user query logs. Cui et al., (2002) developed a system to analyse user logs data to discover the correlation between query terms and document terms. These extracted term correlations were then used for selecting additional terms for query expansion. Their work assumed that the documents visited by the user were relevant to their query. They maintained a list of all the documents visited for a particular query. The probability of a document being visited for a particular query word was calculated to find the relevance of the document. The utilization of recorded user logs was found to improve the accuracy of the retrieval system.

One type of query expansion is to identify phrases within the query. In the work of Cui et al., (2002), involving locating all the phrases in the query space, extracted by finding all sequences of N-grams, where N is the number of nontrivial terms in a query from the user logs with occurrences higher than 5. These N-grams were treated as candidate phrases in the document corpus and those not appearing in the documents were filtered out. They created a thesaurus containing over 13,000 phrases, which were used as additional indexing units. The use of phrases in their system outperformed the use of single terms for query expansion.

Principle of Using User Logs

Since there has been various work with the usage of user logs information to enhance the search accuracy, such as Beeferman and Berger (2000) exploiting “click-through data” in clustering URLs and queries using a graph-based iterative clustering technique, and Wen et al. (2002)

applying FAQs to improve the effectiveness of question answering system. This work indicates that user logs can be mined to identify good indicators of the user interests.

Based on this previous work, Cui et al. (2002) suggested an approach to extract relationships between query terms and documents terms, and to use these correlations to compute the document relevance. They introduced ‘query sessions’ to present the relation between queries and documents in user logs, which are also used to bridge the gap between query terms and document terms. According to Cui et al. (2002): *if queries containing one term often lead to the selection of documents containing another term, then there is a strong relation between the two terms*. Also they define query sessions as follows:

$$\text{query session} := \langle \text{query text} \rangle [\text{clicked document}] *$$

The following sections review the details of the log-based query expansion method proposed in Cui et al. (2002).

Correlations between Query Terms and Document Terms

As shown in Figure 2.8, Cui et al. (2002) build query sessions based on collected user logs. Links are created on both sides of query sessions. Link weights are computed between query terms and queries, also between documents and each document’s terms. Then, the correlation between query terms and document terms can be extracted based on the weights of the links constituting the path between them. Finally, a probabilistic correlations matrix between the terms can be obtained based on the large number of user logs.

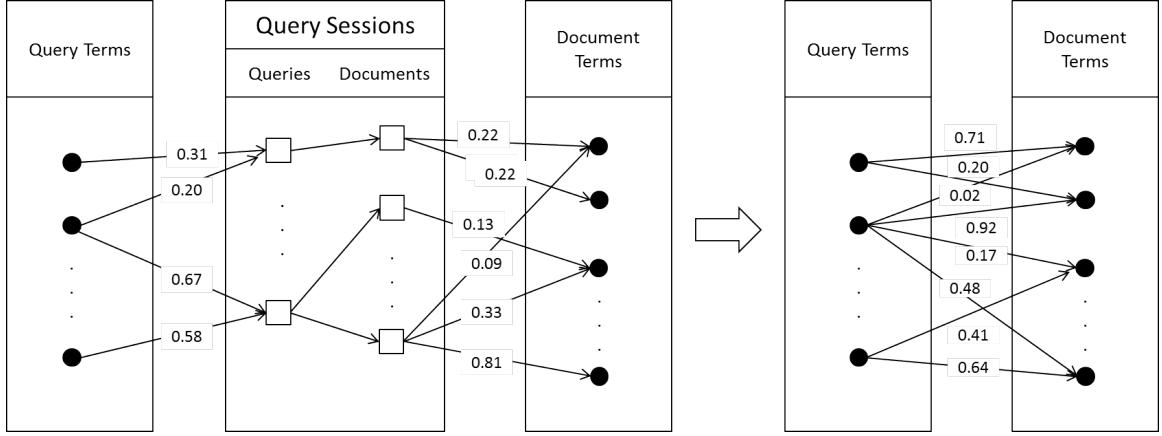


Figure 2.8 Establishing correlations between query terms and documents via query sessions (Cui et al., 2002).

Following are the notation and details of methods introduced in Cui et al. (2002).

In order to determine the degree of correlation between terms, these degrees are defined as the conditional probabilities between terms, i.e., $P(w_j^{(d)} | w_i^{(q)})$ for any document term $w_j^{(d)}$ and any query term $w_i^{(q)}$. The probability $P(w_j^{(d)} | w_i^{(q)})$ can be determined using Equation (2-9).

$$\begin{aligned}
 P(w_j^{(d)} | w_i^{(q)}) &= \frac{P(w_j^{(d)}, w_i^{(q)})}{P(w_i^{(q)})} = \frac{\sum_{\forall D_k \in S} P(w_j^{(d)}, w_i^{(q)}, D_k)}{P(w_i^{(q)})} \\
 &= \frac{\sum_{\forall D_k \in S} P(w_j^{(d)} | w_i^{(q)}, D_k) \times P(w_i^{(q)}, D_k)}{P(w_i^{(q)})}
 \end{aligned} \tag{2-9}$$

where S is a set of clicked documents for queries containing the query term $w_i^{(q)}$. Assume that $P(w_j^{(d)} | w_i^{(q)}, D_k) = P(w_j^{(d)} | D_k)$, which means that the document D_k separates the query term from the document term $w_j^{(d)}$. Thus, $P(w_j^{(d)} | w_i^{(q)})$ can be expressed as shown in Equation (2-10).

$$\begin{aligned}
P(w_j^{(d)} | w_i^{(q)}) &= \frac{\sum_{\forall D_k \in S} P(w_j^{(d)} | D_k) \times P(D_k | w_i^{(q)}) \times P(w_i^{(q)})}{P(w_i^{(q)})} \\
&= \sum_{\forall D_k \in S} P(w_j^{(d)} | D_k) \times P(D_k | w_i^{(q)})
\end{aligned} \tag{2-10}$$

$P(D_k | w_i^{(q)})$ is the conditional probability of the document D_k being clicked when $w_i^{(q)}$ appears in the user's query, $P(w_j^{(d)} | D_k)$ is the conditional probability of occurrence of $w_j^{(d)}$ if the document is selected, $P(D_k | w_i^{(q)})$ and $P(w_j^{(d)} | D_k)$ can be estimated respectively, from the user logs and from the frequency of occurrences of terms in documents as shown in Equation (2-11) and Equation (2-12).

$$P(D_k | w_i^{(q)}) = \frac{f_{ik}^{(q)}(w_i^{(q)}, D_k)}{f^{(q)}(w_i^{(q)})} \tag{2-11}$$

$$P(w_j^{(d)} | D_k) = \frac{w_{jk}^{(d)}}{\sum_{\forall t \in D_k} w_{tk}^{(d)}} \tag{2-12}$$

where

- $f_{ik}^{(q)}(w_i^{(q)}, D_k)$ is the number of query sessions in which the query term $w_i^{(q)}$ and the document D_k appear together.
- $f^{(q)}(w_i^{(q)})$ is the number of query sessions that contain the term $w_i^{(q)}$.
- $P(w_j^{(d)} | D_k)$ is the normalized weight of the term $w_j^{(d)}$ in the document D_k , which is divided by the sum of all term weights in the document D_k .

By combining Equation (2-10), Equation (2-11) and Equation (2-12), the formula for $P(w_j^{(d)} | w_i^{(q)})$ can be obtained as shown in Equation (2-13).

$$P(w_j^{(d)} | w_i^{(q)}) = \sum_{\forall D_k \in S} (P(w_j^{(d)} | D_k) \times \frac{f_{ik}^{(q)}(w_i^{(q)}, D_k)}{f^{(q)}(w_i^{(q)})}) \tag{2-13}$$

Query Expansion Based on Term Correlations

Equation (2-13) describes the calculation of the chance of a single document term being selected as the expansion term for a given single query term. However in order to determine whether this document term is a good expansion term, we need to consider the relationship of this document term to the whole given query.

For this, an idea similar to that of Qiu and Frei (1993) is used by Cui et al., (2002), i.e., expansion terms are selected according to their relationship to the whole query. The relationship between a document term to the whole query is measured by the cohesion calculation which is shown in Equation (2-14).

$$CoWeight_Q(w_j^{(d)}) = \ln \left(\prod_{w_i^{(q)} \in Q} (P(w_j^{(d)} | w_i^{(q)}) + 1) \right) \quad (2-14)$$

which combines the relationships of a particular query term to the rest of the query terms.

In summary, log-based query expansion takes the following steps to expand a new query Q :

- 1 Extract all query terms (eliminating stopwords) from Q .
- 2 Find all documents related to any query term in query sessions.
- 3 For each document term in these documents, use Equation (2-14) to calculate its evidence of being selected as an expansion term according to the whole query.
- 4 Select n document terms with the highest cohesion weight and formulate the new query Q' by adding these terms into Q .
- 5 Use Q' to retrieve documents in a searching system.

In Cui et al., (2002), they show that this log-based query expansion method is an effective way to narrow the gap between the query and document. For new queries, high-quality expansion terms can be selected from the documents on the basis of the extracted term

correlations. The experimental results they obtained also showed that this method can achieve substantial improvements in performance on both long queries and short queries.

Local Methods

In contrast to global analysis, local analysis uses only a subset of retrieved documents for a given query to perform query expansion. Local analysis techniques can be grouped into two categories: based on user feedback information and methods based on information derived from a subset of the retrieved documents. Both of these two categories use different type of relevance feedback information to perform the query expansion.

Relevance Feedback

Relevance Feedback (RF) provides a mechanism for a specific type of query expansion. In RF, the user query is used to retrieve an initial list of ranked documents. The user judges the relevance of the retrieved documents. These judgments are then used by RF to refine the query by adding or potentially removing terms, and a new ranked list is produced using the refined query. The above process can take place iteratively for multiple cycles of retrieval. The actual algorithm employed to refine the query based on RF depends on the underlying retrieval model used.

Based on the way in which feedback information is gathered, RF can be categorized as:

- **Explicit Feedback:** Explicit feedback is obtained from human assessors indicating the relevance of a document retrieved for a query. This type of feedback is defined as explicit only when the assessors or other users of a system know that the feedback provided can be interpreted as relevance judgments.

Users may indicate relevance explicitly using a binary or graded relevance system. Binary relevance feedback indicates that a document is either relevant or

irrelevant for a given query. Graded relevance feedback indicates the relevance of a document to a query on a scale using numbers, letters or descriptions (such as, “not relevant”, “somewhat relevant”, “relevant” or “very relevant”). Graded relevance may also take the form of a cardinal ordering of documents created by an assessor, that is, the assessor places the documents of a result set in order of relevance. An example of this would be the SearchWiki feature implemented by Google on their search website. The RF information needs to be combined with the original query to improve retrieval performance, as is done in the well-know Rocchio Algorithm which is a classic algorithm for incorporating relevance feedback into the Vector Space Model (VSM), introduced by Salton’s SMART system (Rocchio, 1971).

- **Implicit Feedback:** Implicit feedback is inferred from user behaviour, such as noting which documents the user selects or does not select for viewing, the duration of time spent viewing a document, or page browsing or scrolling actions (Xu and Croft, 1996). The key difference between implicit feedback and explicit feedback is that: the user in case of implicit feedback does not assess relevance for the benefit of the IR system, but only satisfy their own needs. An example of this is *Surf Canyon* (Hardtke et al., 2009) which is a computer software company. Both their browser extension and website⁷ use implicit RF in real time to personalize search. Their advanced search results from later pages of the result set are based on both user interaction and time spent viewing the page linked to in a search result.
- **Pseudo Relevance Feedback (PRF):** In PRF, the top k ranked initial retrieved documents are assumed to be relevant and are used to refine the query. The basis

⁷ <http://www.surfcanyon.com/>

behind PRF is that documents which are similar to the user's initial query will lead to more relevant terms, which when augmented with the query will lead to an improvement in performance. Croft and Harper (1979) first suggested this technique for addressing one fundamental problem – query drift. Query drift is caused as a result of adding terms which have no association with the topic of relevance of the query. This happens when there are only a few or no relevant documents in the top k feedback documents. Due to this sensitivity to the quality of top k documents, PRF only improves the performance of queries which have good or reasonable initial retrieval performance. It cannot be used to improve the performance of bad or failed queries which do not retrieve anything relevant initially. Hence, it is not robust to the quality of the initial retrieval.

Several approaches have been proposed to improve the robustness of PRF. The main strategies used can be categorized as follow:

- Jing and Croft (1996) used local and global document analysis to perform query expansion. They compare the retrieval effectiveness of three automatic query expansion techniques: global analysis, local feedback and local context analysis. Their results show that local context analysis, which uses some global analysis techniques on the local document set outperforms both simple local feedback and global document analysis in terms of retrieval effectiveness and predictability.
- Lam-Adesina and Jones (2001) describe an investigation of using document summarization to improve term selection in query expansion for PRF. Retrieved documents are summarized and expansion terms selected from the summaries rather than the whole document. They explored both context-

independent summarization and query-biased summarization. Retrieval results of this method using the TREC-8 ad hoc task show that use of summarization can improve the effectiveness of PRF.

- To refine the feedback document set so that instead of using all the top k documents, a system could choose only a subset of it which appears to have potential of relevant to query (Mitra et al., 1998) (Sakai et al., 2001). Instead of using the same values of k and t (the number of terms added to the initial query) for all test requests, Sakai et al. propose methods for estimating the best PRF parameter values for each test request: Firstly, for each training request, they choose the optimized parameters for it. Secondly, they compute the distance between each training request and test request. Then, for each test request, they choose closest training request and re-use the training request's parameter for this test request. That is, the PRF parameter values for a test request are selected based on a "similar request that the system has seen before".
- Instead of using all the terms obtained through feedback for query refinement, a system might only use a subset of important terms to avoid introducing query drift. In Cao et al. (2008). They propose to use a supervised learning method for term selection, they consider the term selection problem as a term classification problem by attempting to separate good expansion terms from the others directly according to their potential impact on the retrieval effectiveness.
- Dynamically deciding when to apply PRF instead of using it for all queries (Amati et al., 2004). The authors define two information theoretic functions

to predict the query difficulty and selectively apply the query expansion. They avoid the application of query expansion on a set of difficult topics, The query expansion application predictor achieves a performance similar to that of the unexpanded method on the worst topics, and better performance than full query expansion on the whole set of topics.

- Varying the importance of each feedback document, (Tao and Zhai, 2006), present a method for pseudo feedback based on statistical language models. Their main idea is to integrate the original query with feedback documents in a single probabilistic mixture model and regularize the estimation of the language model parameters in the model so that the information in the feedback documents can be gradually added to the original query.
- Use a large external collection like Wikipedia or the web as a source of expansion terms is a good way to perform PRF (Voorhees, 2006) (Xu et al., 2009). For example, the work in Xu et al., they present a systematic exploration of the utilization of Wikipedia in PRF for query dependent expansion. Voorhees (2006) examines the utility of lexical query expansion in a TREC collection. Concepts are represented by WordNet synonym sets and are expanded by following the typed links included in WordNet.
- Lv and Zhai (2010) proposed a positional relevance model where the terms in the document which are nearer to the query terms are assigned more weight. They present an intuition that since topically related content is usually grouped together in text documents, terms closer to the occurrences of query words are, in general, more likely to be relevant to the query topic. Based on this theory, they propose a positional relevance model (PRM) to

incorporate the cues of term positions and term proximity in a probabilistic feedback model based on statistical language modelling.

2.1.4.2 Bo1 Pseudo-Relevance Feedback Query Expansion Methods

Our experiments described in the later chapters of this thesis use the Terrier search engine to perform retrieval. These experiments all utilize the Terrier default Bo1 PRF query expansion method. PRF query expansion on the initial retrieval ranking list, using the top N documents in the initial retrieval list, adds n terms to the current query.

Terrier provides various models for query expansion. We carried out our experiments using the default Bo1 approach, since it uses the effective divergence from randomness term weighting model, which is based on Bose-Einstein statistics and is similar to Rocchio (Amati, 2003) (Macdonald et al., 2005) (Plachouras et al., 2004). Using this model, the weight $\omega(t)$ of a term t in the top-ranked documents is given by Equation (2-15).

$$\omega(t) = tf_t \cdot \log_2 \frac{1 + P_n}{P_n} + \log_2(1 + P_n) \quad (2-15)$$

where tf_t is the frequency of the term t in the pseudo-relevant set (top N document set) and P_n is given by $F/|S|$. F is the term frequency of the query term in the whole collection and $|S|$ is the number of the documents in the collection.

2.1.5 Personalized Search

Besides the short query problem described in Section 2.1, another challenge for IR systems is that if different users submit the same request to the system, they will obtain the same list of results, regardless of their personal interests or experience level. To address this challenge, Personalized Information Retrieval (PIR) has gained much attention in recent years (Agichtein et

al., 2006) (Brusilovsky and Tasso, 2004) (Gauch et al., 2007) (Sugiyama et al., 2004). PIR provides a personalized search service for individual users which seeks to help them in satisfying their information needs more efficiently (Agichtein et al., 2006). PIR systems retrieve documents which are not only relevant to the query itself, but also relevant to the user's interests: as a result of which different users receive different results for the same query, often in different rank order. PIR systems generally use three stages in order to provide their personalized service. These are illustrated in Figure 2.9.

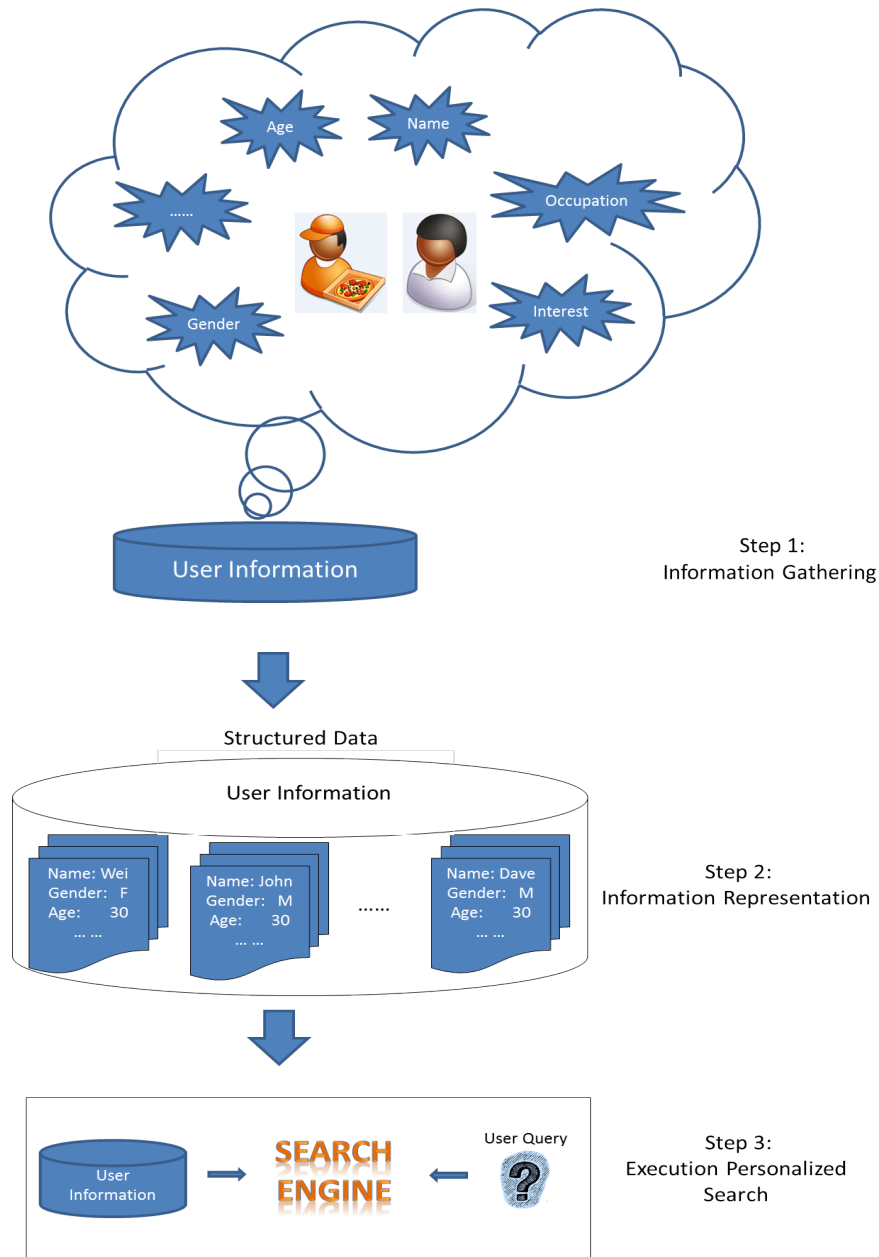


Figure 2.9 Three stages of PIR system provide personalized service

In order to provide users with more personalized results, information gathering is a key stage in the process. Information can be gathered in an implicit manner, where it is obtained without any extra effort from the user, or in an explicit manner, where the user has to explicitly supply information to the system. Explicit information gathering involves collection of user information

about the users themselves, such as their personal details or search interests, and explicit feedback on the relevance or interest to the user of previous search results. An example of gathering explicit feedback is as follows. A user submits a query, the retrieval system performs an initial run to rank documents and presents a few top ranked documents to this user to explicitly judge their relevance. When it has obtained the user relevance judgments of these top ranked documents, the retrieval system combines these judgments with the original query to perform query expansion. It then performs a second run and presents a new ranked document list to the user. Implicit gathering is concerned with gathering information about the users and their usage behaviour when interacting with the system without explicit input from the user, for example, query logs or browsing history. This is carried out by a user submitting a query. The retrieval system performs the first run to rank documents and presents a few top ranked documents. These top ranked documents are assumed to be relevant to the query by the retrieval system and are combined with the original query to perform query expansion. A second retrieval run is then carried out and a new ranked list presented to this user.

PIR techniques have been widely used in popular search engines and work efficiently (Agichtein et al., 2006) (Teevan et al., 2005) (Speretta and Gauch, 2005). In our research, we seek to address the problem that arises in the situation where a user query is on a new topic. This means that it is not possible for us to gather explicit or even implicit information from this user with respect to this topic. Under this condition, we only can gain and use information from other users who have entered queries on similar topics. In order to provide users more relevant results, we attempt to exploit information from the search history of a number of users with similar interests to help refine the query of the current user and use the refined reformulated query to obtain better retrieval results for the current user.

2.2 Recommender Systems

Recommender systems are a subclass of information filtering system which seeks to predict the rating or preference that a user would give to an item. Compared to an IR system, a recommender system not only focuses on discovering items which may satisfy a user's information need, but also to discover novel items which may not be retrieved as potentially relevant to the user's query by an IR system, but which the user may find of interest since they have been selected by other users for queries similar to the current one.

In this section, we review the foundations and use of recommender systems (RSs). The recommender algorithms introduced in this section are exploited in our experimental work described later in this thesis which seeks to improve retrieval effectiveness in the situation where a user enters a query on a new topic.

2.2.1 Background

Recommender systems are generally classified into two categories: content-based filtering (CBF) and collaborative filtering (CF). In general, CBF methods analyse a set of documents rated by an individual user, and use the contents of the documents, as well as the user's ratings of viewed items, to infer a user profile that can be used to recommend additional items of interest. By contrast, the CF approach uses an information filtering technique based on the user's previous evaluation of items by multiple previous users or the history of previous behaviour by the users. The following subsections introduce the advantages and disadvantages of CBF and CF and the methods used to implement them.

2.2.1.1 Content-Based Filtering Method

Systems implementing a CBF approach analyse a set of documents and descriptions of items previously rated by a user and build a model or profile of user interests based on the features of the objects rated by that user (Mladenic and Stefan, 1999). The profile is a structured representation of user interests used to recommend new items which may be of interest based on previous behaviour. The recommendation process basically consists of matching the attributes of the user profile against the attributes of a content object. The result is a relevance judgment that represents the user's level of interest in that object. If the profile accurately reflects the users' preferences, it has tremendous potential to improve the effectiveness of an information access process. For instance, it could be used to filter search results by deciding whether a user is likely to be interested in a specific document or not.

The adoption of a content-based recommendation paradigm has several advantages when compared to the CF methods:

- User independence - Content-based recommenders solely exploit ratings provided by the current user to build a user's own profile. Instead, CF methods need ratings from other users in order to find the 'nearest neighbours' of the current user, i.e., users that have similar tastes since they have rated same items similarly. Only items that are preferred by the neighbours of the active user will be recommended.
- Transparency – Explanations on how the recommender system works can be provided by explicitly listing content features or descriptions that caused an item to appear in the list of recommendations. These features are indicators to consult in order to decide whether to trust a recommendation. Conversely, CF systems are black boxes since the only explanation for an item recommendation is that unknown users with similar preferences liked that item.

- New item – Content-based recommenders are capable of recommending items not yet rated by any user. As a consequence, they do not suffer from the first-rater problem, which affects CF recommenders relying solely on the users' preferences to make recommendations. Therefore, until the new item has been rated by a substantial number of users, the system is not able to recommend it.

Nonetheless, content-based systems also have several shortcomings:

- Limited content analysis – Since content-based techniques are limited by the features that are associated with the objects that these systems recommend. Therefore, in order to have a sufficient set of features, the content must either be in a form that can be parsed automatically by a computer or the features should be assigned to items manually. While IR techniques work well in extracting features from text documents, some other domains may have problems with automatic feature extraction, such as multimedia data. Moreover, it is often not practical to assign features manually because of the limitation of resources to achieve this. At least in part, content-based filtering is insufficient to deal with much of the information available today.
- Over-specialization - Content-based recommenders have no inherent method for finding something unexpected. The system suggests items whose scores are high when matched against the user's profile, hence the user is recommended items similar to those they already rated. This drawback is also called the “serendipity problem”, to indicate the tendency of content-based systems to produce recommendations with a limited degree of novelty. To give an example, when a user has only rated movies directed by Stanley Kubrick, the user will be recommended only this kind of movie. A ‘perfect’ content-based technique would rarely find anything novel, which naturally limits the range of applications for which it would be useful.

- New user - Enough ratings have to be collected before a content-based recommender system can really understand current user's preferences and provide accurate recommendations for this user. Therefore, when few ratings are available for current user, as for a new user, the system will not be able to provide reliable recommendations (Pasquale et al., 2011).

2.2.1.2 Collaborative Filtering Method

Since Collaborative filtering (CF) systems (Shardanand and Maes, 1995) are not based on item content, they are less sensitive than content-based filtering methods. CF systems provide current user's recommendation based on the ratings provided by other users who share similar interests. This makes CF systems valid for any type of item content.

In CF-based systems, users profile information is a set of items and their corresponding ratings obtained from users. User's rating information is usually collected in two ways: explicitly, asking users to provide ratings for given items; and implicitly, by recording users' behaviour. Generally, three types of ratings are presented: unary value, binary value and numerical value on a finite scale. Depending on how the data of the rating is processed, two types of algorithms, memory-based (also called neighbour-based) and model-based can be differentiated.

Memory-based algorithms use all users' ratings to compute the predicted rating. Generally, they compute the similarity between each previous user with the current user, and select the closest neighbours for current user. Finally, prediction is conducted on the basis of the rating data of these identified similar neighbours. Most of these algorithms can be classified as user-based algorithms or item-based algorithms, depending on whether the process of getting

neighbours is focused on finding similar users (Resnick et al., 1994) (Shardanand and Maes, 1995) or items (Sarwar et al., 2001).

Model-based algorithms first construct a model to represent the behaviour of the users, to predict the ratings (Cacheda et al., 2011). The parameters of the model are estimated offline using data from the rating matrix. Multiple approaches to this have been reported in the literature, including those related to machine learning (Marlin, 2004): linear algebra (SVD (Billsus and Pazzani, 1998) (Sarwar et al. 2001)), factor analysis (Canny, 2002), clustering (Ungar and Foster, 1998) (Kohrs and Merialdo, 1999), neural networks (Billsus and Pazzani, 1998), graphs (Aggarwal et al., 1999), probabilistic methods, such as Bayes networks (Breese et al., 1998) and latent class models (Si and Jin, 2003) (Hofmann, 2004).

Generally, memory-based algorithms are simpler than other recommender algorithms, and obtain reasonably accurate results (Delgado and Ishii, 1999). However, they are much more sensitive than model-based algorithms to some common problems of recommender systems. In Cacheda et al. (2011), the authors highlight the following drawbacks of memory-based algorithms:

- Sparsity of the rating matrix (Sarwar et al., 2001; Huang et al., 2004) - In most recommender systems, each user rates only a small subset of the available items, so most of the cells in the rating matrix are empty. In such cases, finding similarities among different users or items is challenging (Cacheda et al., 2011).
- Cold-start (Schein et al., 2002) - Related to the previous problem, there are two kinds of cold-start problem: the user side cold-start problem and the item-side cold start problem. The user cold-start problem relates to the difficulty of making recommendations for users recently introduced into the system. In such a case, the new user has not yet rated enough items, so the recommender system is unable to determine their interests. Some systems

overcome this problem by forcing the user to first rate a given set of items. However, this is generally not possible since most users are reluctant to do this, also these initial ratings can introduce biases into the system because the user's interests change frequently. Secondly, the cold-start problem also affects new items, since they will not be recommended until enough users have rated them, which is called the item side cold-start problem (Cacheda et al., 2011).

- Shilling⁸ (Lam and Riedl, 2004) (Chirita et al., 2005) - Recommender systems can suffer spam attacks, mainly from users interested in misleading the system to recommend a certain product. Several techniques affecting both memory-based (Mobasher et al., 2007) and model-based (Sandvig et al., 2007) algorithms have been studied in the past (Cacheda et al., 2011).

These problem are partly reduced by model-based algorithms, however, they still present several problems. Many of them are extremely complex, as they have a multitude of parameters to estimate, and frequently they are sensitive to data changes (Cacheda et al., 2011). Moreover, model construction (and update, when new data is added) usually takes a long time. To avoid these problems, some researchers have developed algorithms that use models that are fast and easy to calculate (Lemire and Maclachlan, 2005). In recent years, algorithms based on a variation of singular value decomposition (SVD), where only known ratings are modelled, have become very popular (Paterek, 2007) (Funk, 2006). These can be trained easily using a simple gradient descent technique to achieve both good accuracy and efficiency. Other researchers have combined techniques from both model-based and memory-based algorithms (Pennock et al.,

⁸ Lam and Riedl (2004) introduce a Shilling Recommender System to detect and filter the shill attack in e-commerce.

2000) to take advantage of the best of both worlds. For example, Koren (2009) obtained good results by integrating a regularized SVD model with a model inspired by neighbour-based algorithms (Cacheda et al., 2011).

The hybrid approach is described in the next section. This method combines content-based filtering with collaborative filtering to improve the effectiveness of recommender systems.

2.2.1.3 Hybrid Method

Recent research has demonstrated that a hybrid approach, combining collaborative filtering and content-based filtering can be more effective in some cases than either technique on its own. Hybrid approaches can be implemented in several ways: by making CBF and CF predictions separately and then combining them; by adding CBF capabilities to a CF approach (and vice versa); or by unifying the approaches into a single model. Several studies have empirically compared the performance of hybrid methods with pure collaborative and CBF methods, and demonstrate that hybrid methods can provide more accurate recommendations than individual approaches. These methods can also be used to reduce the effect of some of the common problems in recommender systems such as the cold start and sparsity problems. For instance, when systems suffer from the data sparsity problem, there is not enough other user rating information available for CF algorithms to make recommendations. The CBF method can help to improve the results by comparing the interests of the current user to each item based on content information. For the item side cold start problem, since new items have not yet been rated by users, the CBF method can again help to improve the recommendation results based on the new item's content. However, the user side cold-start problem is still a challenge for current RSs, because both the CBF and CF methods need the current user to rate enough items in order to learn their interests.

The user side cold start problem is also a challenge we consider in this thesis. The method introduced later in this thesis combines IR with RSs, while the RS's output is not reliable when suffering from this problem, the IR component can still obtain reasonable results in order to provide competitive results for users.

2.2.2 Recommender Algorithms

Following the introduction to recommender systems in the previous sections, in this section, we overview some of the most popular content-based algorithms and collaborative filtering (CF) algorithms.

2.2.2.1 Content-Based Algorithms

Most content-based recommender algorithms use relatively simple retrieval models, such as keyword matching or the Vector Space Model (VSM) with basic TF-IDF (Term Frequency-Inverse Document Frequency) weighting.

Document representation in the VSM raises two issues: weighting the terms and measuring the feature vector similarity. The most commonly used term weighting scheme, TF-IDF weighting, is based on empirical observations regarding the nature of text (Salton, 1989):

- Rare terms are more valuable than frequent terms (IDF assumption) across a collection of documents.
- Multiple occurrences of a term in an individual document are more valuable than single occurrences (TF assumption)
- Long documents are not preferred to short documents (normalization assumption)

In other words, terms that occur frequently in one document but rarely in the rest of a corpus, are more likely to be significant in describing the topic of the document. In addition, normalizing

the resulting weight vectors prevents longer documents from having a greater chance of retrieval at high rank. These assumptions are exemplified by the TF-IDF function shown in Equation (2-16).

$$TF - IDF(t_k, d_j) = \underbrace{TF(t_k, d_j)}_{TF} \cdot \underbrace{\log \frac{N}{n_k}}_{IDF} \quad (2-16)$$

where N denotes the number of documents in the corpus, and n_k denotes the number of documents in the corpus in which the term t_k occurs at least once. $TF(t_k, d_j)$ is computed as shown in Equation (2-17).

$$TF(t_k, d_j) = \frac{f_{k,j}}{\max_z f_{z,j}} \quad (2-17)$$

where the maximum is computed over the frequencies $f_{z,j}$ of all terms t_z that occur in document d_j . In order for the weights to fall in the $[0,1]$ interval and for the documents to be represented by vectors of equal length, weights obtained using Equation (2-16) are usually normalized using cosine normalization as shown in Equation (2-18).

$$w_{k,j} = \frac{TF - IDF(t_k, d_j)}{\sqrt{\sum_{s=1}^{|T|} (TF - IDF(t_s, d_j))^2}} \quad (2-18)$$

As stated earlier, a similarity measure is required to determine the closeness between two documents. Many similarity measures have been derived to describe the proximity of two vectors; among these measures, cosine similarity shown in Equation (2-19) is the most widely used.

$$sim(d_i, d_j) = \frac{\sum_k w_{ki} \cdot w_{kj}}{\sqrt{\sum_k w_{ki}^2} \cdot \sqrt{\sum_k w_{kj}^2}} \quad (2-19)$$

In content-based recommender systems relying on the VSM, both user profiles and items are represented as weighted term vectors. Predictions of a user's interest in a particular item can be derived by computing the cosine similarity.

2.2.2.2 Collaborative Filtering Algorithms

This section describes some of the existing CF algorithms which are employed in our experiments in later chapters. The section begins by identifying the notations used to denote the variables in the description of each CF algorithm.

Notations

CF techniques, as a type of recommender system, aim to recommend useful items to users. In the most typical scenario, these techniques deal with a set of users. Cacheda et al. (2011) define the variables as follows, in order to describe the algorithms more easily, we follow the same notations.

$U = \{u_1, u_2, \dots, u_m\}$ denotes a set of users, and $I = \{i_1, i_2, \dots, i_m\}$ a set of items. Each user, $u_i \in U$ has an associated profile consisting of the subset of items that user u has rated, $I_u \subseteq I$, and the corresponding rating for each item. Similarly, the subset of users that have rated a certain item is defined, $U_i \subseteq U$. The current user, the user for whom a prediction is being obtained, is denoted as u_a . The ratings usually correspond to integer numbers in a certain range, being R , the set of possible ratings.

The matrix V , representing the user ratings, is defined using user profiles. Each element of V , $v_{ui} \in R \cup \emptyset$, denotes the rating given by user $u \in U$ to item $i \in I$, the value \emptyset indicating that the user has not yet rated the item.

In fact, the objective of the CF algorithm is to predict the value of v in these cases. We denote as $p_{ui} \in R \cup \emptyset$, the prediction that the algorithm makes for the rating of item $i \in I$ by user $u \in U$. If the algorithm is not able to make this prediction, $p_{ui} = \emptyset$.

Finally, we define the subset of user ratings, $v_u = \{v_{ui} \in V / i \in I_u\}$, and the subset of item ratings, $v_i = \{v_{ui} \in V / u \in U_i\}$. We denote the user mean rating as $\overline{v_u}$ and the item mean rating as $\overline{v_i}$.

• Item-Based Filtering Algorithm

The item-based approach looks at the set of items that the current user has rated and computes how similar they are to the other items which have not been rated by the current user, and then selects a number of most similar items. Once the most similar items have been found, the prediction is computed by taking a weighted average of the current user's ratings on these similar items. The item-based filtering method contains two steps:

1. Build an item-item matrix determining relationships (similarity) between pairs of items.
2. Using the similarity matrix, and the data on the current user, infer the current user's taste.

Different strategies can be used as a similarity measure, although Sarwar et al. (2001) has concluded that the adjusted cosine similarity, Equation (2-19), obtains the best results by far. We thus choose this method in this thesis.

$$S(i, j) = \frac{\sum_{u \in U} (v_{ui} - \overline{v_u})(v_{uj} - \overline{v_u})}{\sqrt{\sum_{u \in U} (v_{ui} - \overline{v_u})^2 \sum_{u \in U} (v_{uj} - \overline{v_u})^2}} \quad (2-19)$$

After calculating the similarity between items, N most similar neighbours are selected, and their ratings are used to compute the prediction weight for target item using Equation (2-20).

$$p_{aj} = \frac{\sum_i (s(j, i) v_{ai})}{\sum_i |s(j, i)|} \quad (2-20)$$

In item-based method, the similarity between items can be computed offline, since this is more stable than the similarity between users.

- **Cluster-Based Smoothing**

A cluster is a group which contains a number of similar objects. Cluster analysis or clustering is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar to each other than to those in other groups (clusters).

The cluster-based smoothing method groups users to reach two objectives: to increase the density of the rating matrix, and to increase the scalability. The details of this algorithm can be found in (Xue et al., 2005).

- 1 Offline, the users are grouped into n clusters, using a K-means algorithm.
- 2 The cluster mean is used to fill the user profile, that is, for the items that a user has not rated, the rating is set as the mean of the ratings of the other users in the cluster.
- 3 The clusters closest to the active user's cluster are selected.
- 4 Other users in the clusters selected as most similar to the active user are used as the neighbours of the active user.
- 5 Finally, the prediction rating for each item which the active user has not rated is set to the weighted mean rating of the active user's neighbours.

- **Tendency-Based Filtering Algorithm**

Most CF algorithms reviewed here are based on similarities among users or items. Although many different techniques have been used to process the data, most focus on finding more or less hidden relationships. As an alternative, Cacheda et al. (2011) propose a tendency-based CF algorithm. Instead of looking for similarity relations between users and items, this approach looks at the differences between them. Cacheda et al. suggest that users rate items in different ways, variations are related to their differences in opinions and tastes. Besides this, they also observe that even when users have similar preferences, they may rate items in different ways: some users are more inclined to give positive ratings, leaving negative ratings for really bad items; others might save their highest ratings for the best items and tend to give negative ratings to all other items. They suggest that these similar taste users with differing approaches to rating are an accurate indicator of the quality of each particular document and its utility for users.

This tendency-based CF algorithm interprets the variations of users to interpret positive and negative opinions differently in two ways: the tendencies of users and items. In (Cacheda et al., 2011), the concept of tendencies of users refers to whether a user tends to rate items positively or, by contrast, negatively. The tendency of a user (τ_u) as the average difference between user's ratings and the item mean is shown in Equation (2-21).

$$\tau_u = \frac{\sum_{i \in I_u} (v_{ui} - \bar{v}_i)}{|I_u|} \quad (2-21)$$

In Cacheda et al., (2011), the tendency of an item, represented by τ_i , captures whether users consider it an especially good or especially bad item. This is calculated using Equation (2-22).

$$\tau_i = \frac{\sum_{u \in U_i} (v_{ui} - \bar{v}_u)}{|U_i|} \quad (2-22)$$

According to the definition of user tendency and item tendency, it can be noted that a tendency-based CF algorithm takes into account the user rating mean and item rating mean when computing a prediction. Several situations can arise when computing a prediction using the tendency-based CF approach, see Figure 2.10.

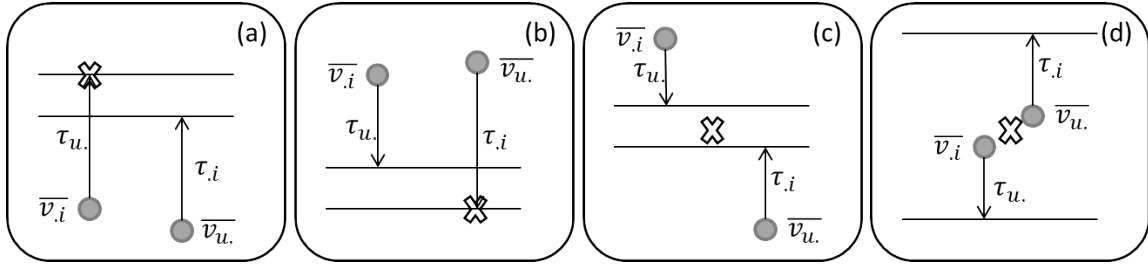


Figure 2.10 Possible relations between means (circles), tendencies (arrows) and prediction (cross) (Cacheda et al., 2011).

1. In the first case, Figure 2.10(a), both the user and the item have a positive tendency, the prediction is computed using Equation (2-23).

$$P_{ui} = \max (\bar{v}_u + \tau_i, \bar{v}_i + \tau_u) \quad (2-23)$$

where the reason for using the maximum is to give a better rating to those items whose tendency indicates that they are good items for the current user.

2. The second case, Figure 2.10(b), is the opposite condition. Both the user and item have a negative tendency. In other words, the user usually rates items below their mean and the item tends to be rated below the user mean. In this case, the prediction is calculated using Equation (2-24).

$$P_{ui} = \min (\bar{v}_u + \tau_i, \bar{v}_i + \tau_u) \quad (2-24)$$

3. The third case, Figure 2.10(c), occurs when we come across a negative user (a user whose tendency is to rate items below their mean), and a good item (its tendency is to be

rated above the user mean), or vice versa⁹. The prediction is computed using Equation (2-25).

$$P_{ui} = \min(\max(\overline{v_u}, (\overline{v_i} + \tau_u) \cdot \beta + (\overline{v_u} - \tau_i)(1 - \beta)), \overline{v_i}) \quad (2-25)$$

where β is a parameter that controls the contribution of the item and user means.

4. Finally, Figure 2.10(d), depicts the situation where the means do not corroborate the tendency, the prediction is computed using Equation (2-26).

$$P_{ui} = \overline{v_i} \cdot \beta + \overline{v_u}(1 - \beta) \quad (2-26)$$

As observed from the four cases introduced above, simple formulae are used in each of the four cases, and the calculation does not depend on the number of users or items in the system.

- **Rating-Based Filtering Algorithm**

Rating-based CF is the process of predicting how a user would rate a given item based on other user ratings. An online rating-based CF query consists of a set of (item, rating) pairs from a single user. The response to this query is a list of predicted (item, rating) pairs for those items which the current user has not yet rated. The Slope One schemes proposed by Lemire and Maclachlan (2005), with predictors of the form $f(x) = x + b$, precompute the average difference between the ratings of one item and another for users who rate both. Slope One schemes have several advantages, they are: easy to implement and maintain; easy to add new ratings; efficient at query time and expect little input from visitors.

⁹ Or vice versa, positive user and bad item.

The Slope One scheme is a family of algorithms used for CF. Its simplicity makes it especially easy to implement efficiently, while its accuracy is often comparable to more complicated and computationally expensive algorithms (Lemire and Maclachlan, 2005).

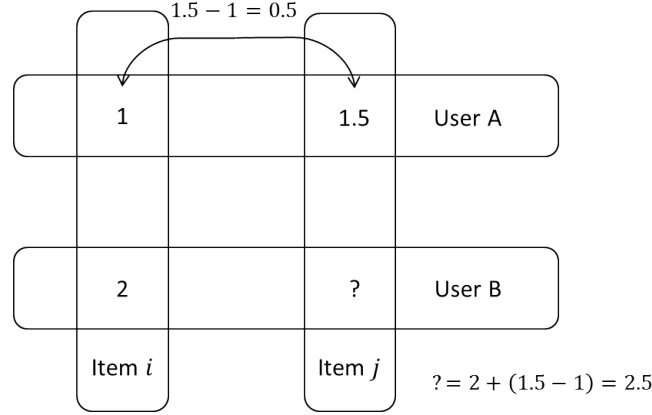


Figure 2.11 Basis of Slope One schemes: User A 's ratings of two items and User B 's rating of a common item is used to predict User B 's unknown rating (Lemire and Maclachlan, 2005).

The Slope One scheme takes into account information both from other users who rate the same items as the current user, and from other items which have been rated by the current user. However, the schemes also rely on data points that fall neither in the user array nor in the item array (e.g. user A 's rating of item i in Figure 2.11).

In Lemire and Maclachlan's paper (2005), they given two evaluation arrays v_i and w_i with $i = 1, \dots, n$, in order to search for the best predictor of the form $f(x) = x + b$, the Slope One algorithm seeks to predict w from v by minimizing $\sum_i (v_i + b - w_i)^2$. Deriving with respect to b and setting the derivation to zero, gives $(\sum_i w_i - v_i)/n$. In other words, the constant b must be chosen to be the average difference between the two arrays. This result motivates the following scheme.

Given a training set S , and any two items j and i with ratings u_j and u_i respectively, in some user evaluation u (annotated as $u \in S_{j,i}$), consider the average deviation of item i with respect to item j shown in Equation (2-27) (Lemire and Maclachlan, 2005).

$$dev_{j,i} = \sum_{u \in S_{j,i}} \frac{u_j - u_i}{card(S_{j,i})} \quad (2-27)$$

where $card(S_{j,i})$ is the number of previous queries receiving a rating for i and j in set S . Note that any user evaluation u not containing both u_j and u_i is not included in the summation. The matrix defined by $dev_{j,i}$ can be computed once and updated quickly when new data is entered.

Given that $dev_{j,i} + u_i$ is a prediction for u_j given u_i , a reasonable predictor might be the average of all such predictions shown in Equation (2-28).

$$P(u)_j = \frac{1}{card(R_j)} \sum_{i \in R_j} (dev_{j,i} + u_i) \quad (2-28)$$

where $R_j = \{i | i \in S(u), i \neq j, card(S_{j,i}) > 0\}$ is the set of all relevant items. An approximation can be used to simplify the calculation of this prediction. For a dense enough dataset where almost all pairs of items have ratings, that is, where $card(S_{j,i}) > 0$ for almost all i, j , most of the time $R_j = S(u)$ for $j \notin S(u)$ and $R_j = S(u) - \{j\}$ when $j \in S(u)$. Since $\bar{u} = \sum_{i \in S(u)} \frac{u_i}{cardS(u)} \simeq \sum_{i \in R_j} \frac{u_i}{card(R_j)}$ for most j , the prediction formula for the Slope One scheme can be simplified to Equation (2-29) (Lemire and Maclachlan, 2005).

$$P^{S1}(u)_j = \bar{u} + \frac{1}{card(R_j)} \sum_{i \in R_j} dev_{j,i} \quad (2-29)$$

The Weighted Slope-One Scheme

One drawback of the basic Slope One algorithm is that the number of ratings observed is not taken into consideration. Intuitively, to address this issue, (Lemire, 2005) proposes a modified Weighted Slope-One prediction shown in Equation (2-30).

$$p^{ws1}(u)_j = \frac{\sum_{i \in S(u) - \{j\}} (dev_{j,i} + u_i) card(S_{j,i})}{\sum_{i \in S(u) - \{j\}} card(S_{j,i})} \quad (2-30)$$

In (Lemire and Maclachlan, (2005), Slope One schemes were compared to other prediction schemes, such as the basic algorithm Per User Average scheme, Bias From Mean (Herlocker et al., 1999) and the memory-based Pearson scheme (Resnick et al., 1994), and also to the item-based approach that is reported to work best in (Breese et al., 1998). Their results indicate that the Weighted Slope-One scheme is reasonably accurate despite its simplicity.

2.3 Integrateing IR with RSs

It has been noted that RSs share fundamental aspects with IR techniques, and that there is some continuity between these two fields of research (Belkin and Croft, 1992). RSs have usually been seen as an IR technique applied where no explicit query has been provided, but a user profile is known instead. However, they were developed independently. Recently, some researchers have started to look for explicit links between them (Breese et al., 1998) (Wang et al., 2008). In this section, we give a brief introduction to some existing related work.

Several attempts have been made to relate CF to IR (text) models, based on which interesting new CF techniques have been derived. Connections to IR model elements can be found in various state-of-the-art CF approaches. Breese et al. (1998) use the IR cosine formula as a similarity function, according to how they rate items, as a special case of vector similarity. This

can be seen as a first attempt to relate memory-based CF with IR. More significantly, in (Soboroff and Nicholas, 2000), the authors present a formal relationship between neighbourhood-based CF and text retrieval by using the generalized vector space model. However, they do not evaluate or implement their approach, and besides, although the model proposed could make use of the rating matrix, they suggest using content-based profiles in order to overcome sparsity, and thus this model would not only depend on explicit ratings. Finally, another problem in which IR and CF have been identified and techniques from the former have been used in the latter is the scalability problem. In (Cöster and Svensson, 2002), the authors proposed the use of inverted indexes for CF in order to speed up the execution time. In this work, various heuristics were used in order to efficiently search in a user's neighbourhood. In this way, memory-based algorithms can perform fast rating predictions by using disk based inverted files. Despite identifying this relationship, no formal theory is presented, and the authors simply cast the CF problem into an IR setting. A similar approach used to personalize IR is proposed in (Pickens et al., 2010). Furthermore, in (Cohen and Lewis, 1999), the use of inverted files is able to accelerate any application involving (sparse) matrix multiplications, which is very common in recommendation techniques, such as matrix factorization methods (Koren et al., 2009).

Based on this previous work which involves IR and CF in the same task, the following sections introduce three of the most recent and effective methods for combining them (Costa and Roda, 2011) (Bellogin and Wang, 2011). This work exploiting the links between IR and RSs seeks to present some methods for re-formulating the RS problem as an IR one. The basic concept of doing this reformulation is to move from the RS domain to the IR one. They consider each user as a document and each item rating by this user as a term: in this way, as in an IR model, a document is a set of terms, and in the RSs field a user is characterized by a set of items to which they have given ratings. Moreover, the active user becomes the query to the IR system,

which means that in the IR system we want documents which are more similar to the query, and for an RS problem we want users who are more similar to the active user. Beyond this point, they can choose any of the existing IR algorithms to obtain the ranking list that represents the set of users more similar to the active user ordered by decreasing similarity. Finally, they use the obtained ranking list to give predictions to the active user, which again goes back to the RS domain.

Text Retrieval Method in Collaborative Filtering

In their work, Bellogin and Wang (2011) unify IR and CF in a way that defines the term frequency and the query in the CF space and then applies these re-defined data in the CF formulae.

- They chose the VSM as the general text retrieval framework. Documents and queries were represented as vectors, so $d^i = [tf(t_1, d^i), \dots, tf(t_T, d^i)]$ is the vector representation for document d^i , while $q = [qf(t_1), \dots, qf(t_T)]$ is the representation for query q .

- Rating-based CF algorithms were used to compute the predicted rating of each document. Letting r_i^u represent the rating assigned to item i by the user u , where $r_i^u \in \{1, \dots, R\}$, assuming a rating scale from 1 to R . The main goal of a rating-based CF system is to predict the user rating for an unknown item, which means finding the most accurate prediction \hat{r}_i^u . This can be formulated in a general way: $\hat{r}_i^u = \sum_{e \in h(u)} f(u, i, e)$, where f and h are depending on the user- or item-based collaborative scheme, and e represents an item.

- Finally taking $qf(t) = w_e^u$ (the rating that user u gives to item e) and $tf(t, d) = w_e^i$ (the similarity between items i and e), they obtained equivalent scoring functions to those defined by standard CF algorithms to predict a rating score for all the items contained in the test set for the current user.

Bellogin and Wang's research applies the standard IR model to an existing CF algorithm to give a recommender algorithm ranking for all items.

Recommender Systems by means of Information Retrieval

Costa and Roda (2011) present a method for reformulating the RS problem as an IR one.

- At first they moved from the RSs domain to the IR one, they considered each user in RSs equal to a document in IR system, and each document in RSs is seen as a term in an IR system: in this way, as in IR, a document is a set of terms, in the RS field a user is characterized as a set of documents (for which the user has given a rating).
- The active user becomes the query in this phase, which means that, while in an IR system we want to identify the documents which are most similar to the query, for the RS problem we want to find the users which are most similar to the active user. At this point Costa and Roda used an existing IR algorithm to obtain a ranking list that represents the set of users most similar to the active user.
- Finally, the ranking list was used to obtain the prediction for the active user.

As mentioned, hybrid approaches which exploit the link between IR and RSs seek to present some methods for re-formulating the RS problem as an IR one.

From the hybrid approached depicted above, we can observe that:

- These hybrid methods either use the IR retrieval model in a RS way or consider the RS in a IR way, which is not a true integration of IR and CF.
- They attempt to use RS to improve the effectiveness of an IR system (or to use an IR system to aid a RS), however, they ignore the fact that RS and IR systems have different goals where an IR system is trying to provide relevant documents to current query, while an RS is attempting to provide novel items to the user. This type of combination may

lead to poor results in some situations when IR or RS challenges occur, for example those relating to short queries or cold start.

Based on these observations, different from all these existing works, the method we propose in this thesis combines IR with RSs together to improve the effectiveness of standard IR results to aid users to find results which meet their current information need. We output the IR and RSs results based on other users' search behaviour history information, and utilize results from a recommender component to re-rank the IR output in a final step. We believe that this approach has the potential to retrieve better results for users compared to a standard IR system.

HeyStaks system

HeyStaks is a system which emphasis the potential for collaboration within Web search as a route to an improved search experience (Smyth et al., 2009), and was originally developed by University College Dublin.

HeyStaks adds two features to a normal search engine: first, it allows a user to build their own search Staks; second, a user's private search Staks can be shared with other users. It also allows other users to add their search experiences into the shared staks, all this information can be used to generate recommendations to refine the search result for users. Smyth et al. (2009) use the example of Google search with HeyStaks to show its effectiveness. Figure 2.13 shows the structure of HeyStaks when using Google search. It contains two components: a client side browser toolbar and a back-end server. The toolbar allows users to create and share Staks and provide them services such "ask tag" and "vote", it also records users' click through data and manages the combination of recommendations with the default Google result list. The backend server manages the individual Stak indexes, the Stak database, the HeyStaks social networking service and the recommendation engine.

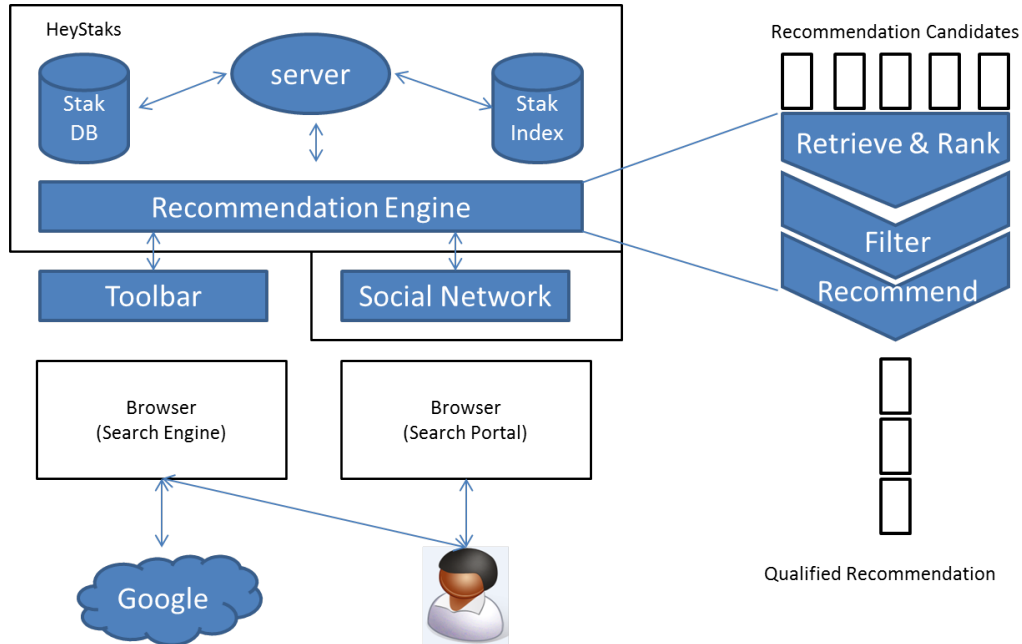


Figure 2.12 They HeyStaks system architecture and outline recommendation model (Smyth et al., 2009).

For the recommendation engine, HeyStaks sets up three steps to generate the recommendations:

Retrieval and Ranking: two types of promotion candidates contribute to the recommendation ranking: primary promotions and the secondary promotions. Primary promotions are the results generated from the active Stak and secondary promotions are the results from the searcher's Stak list.

Evidence-Based Filtering: in order to filter the noise exists in Staks, HeyStaks employs evidence filtering. It sets a variety of threshold models to evaluate the relevance of a particular result, such as the tagging evidence and voting evidence, etc. This noise filtering process is done before making recommendations.

Recommendation Rules: this step decides how to insert the recommendations into the search engine result list. The example of Google search in (Smyth et al. 2009) follows three rules to add recommendations to the default Google result list:

- Add the top 3 primary promotions to the top of the Google result list and label them using the HeyStaks icon.
- Label the remaining primary promotion if they are in the default Google result list.
- Insert the other remaining primary promotions into the secondary promotion list which is sorted by the tf-idf scores. These recommendations are used as the optional, expandable list for the Google search (Smyth et al., 2009).

The active Stak is equal to the topic category we use for the active query in our work. However, different from the HeyStaks, in our experiment we focus on the condition where we do not have any information about the active query, which means that secondary promotions are not available in our task. Also, we add recommendations to the default search result list based on the relevance score they obtain, instead of inserting top recommendations on the top of search result list or labelling them in the existing result list.

2.4 Summary

In this chapter, we reviewed three fields of work: IR systems, RSs and work of examining the combination of IR and RS. For IR systems, we introduced the background to query expansion technology and some existing methods of performing query expansion. We also reviewed personalized search and challenges for this topic. Besides IR systems, various RSs were also reviewed and analyzed. In the third part of the chapter, work exploring the combination of IR

and RS was introduced and the deficiencies of current work on these hybrid approaches was highlighted.

In the next chapter, the basic framework of our proposed integrated IR model is presented and preliminary experiments are described, results of our new approach are compared to the output of a standard IR system.

Following Chapter 3, subsequent chapters further investigate the effectiveness of the integrated IR mode. These studies build on the work reviewed in this chapter to examine the performance of our IR model, based on our results obtained from using existing methods within integrated models, we discover the weaknesses of exploiting these algorithms in our model and propose novel methods to improve their effectiveness in our model.

Chapter 3

An Initial Investigation of an Integrated Information Retrieval- Recommender System Model

In this chapter we describe the basic framework of our proposed integrated information retrieval (IR) model which combines an IR system with a recommender system (RS). An initial experimental study is introduced. These experiments are conducted on simulated user search behaviour based on two different datasets: INEX 2009 and TREC-8. The purpose of these studies is to investigate the potential effectiveness of the integrated IR model. However, since there is no data collection directly available for the experiment, in this chapter, we choose a simulation method to simulate user search behaviour data in different conditions. For the recommender component, a rating-based collaborative filtering (CF) method is chosen to compute the recommendation. The experiments achieve good results illustrating that combining the IR output with RS results has the potential to improve standard IR system results, but with a drawback of being sensitive to noise. In the following chapters, we examine different recommender algorithms in our integrated model and conduct experiments on a real world test collection to further investigate the effectiveness of the model.

3.1 The Framework of Integrated IR Model

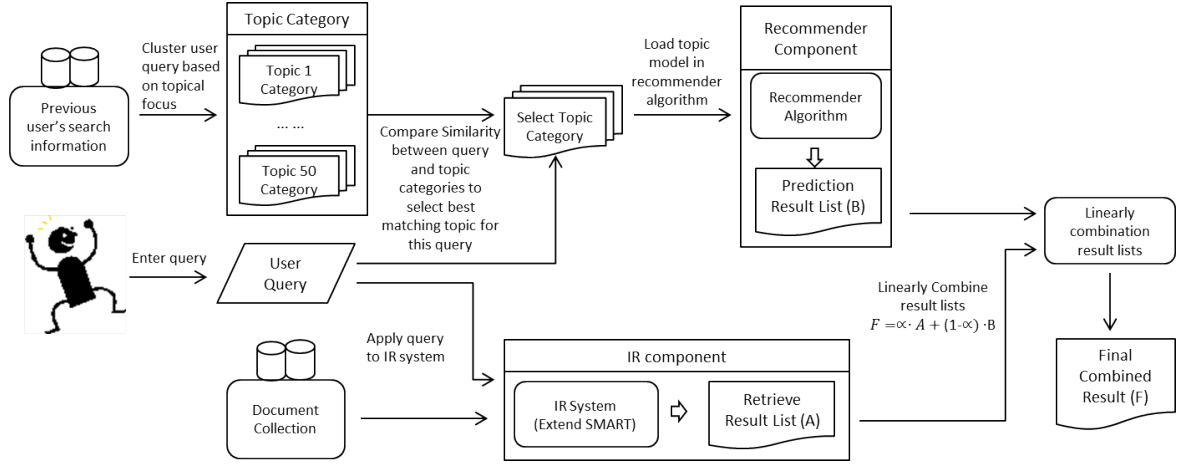


Figure 3.1 Framework of our proposed retrieval model

As introduced in Chapter 2, most existing related work uses IR concepts to reformulate an RS model or the reverse of this using an RS scheme to reformulate an IR model. Our proposed approach is quite different from this existing work. We propose an integrated IR model which combines recommendation results with IR results. We build IR and RS components separately, and carry out the combination in the final step before presentation of the search results. The aim of our integrated IR model is to better address the challenge of short queries from users. Personalized IR (PIR) is widely used to address this challenge in some settings, however in some conditions, personal data for the user relating to their information need is unavailable, such as for a user query on a new topic, or where the user is reluctant to provide personal information related to the topic of the query. In this condition, we attempt to exploit the search behaviour of previous users to aid the current user to find better results to meet his/her information needs. The integrated method employs the rating-based recommender algorithm, which is based on information gathered from other searchers working with related topics, to augment the output of an IR system.

Figure 3.1. shows the framework of our proposed integrated IR model. This method for integrating a recommender component into a standard IR system proceeds as follows:

- The system records information from previous searches. These include each query entered by each user to search the available document archive, implicit feedback from the user for each retrieved document viewed by the users in the form of the time that the searcher spends on viewing each document based on the assumption that the viewing time is correlated with the relevance of the documents.
- The user search sessions are categorized into different topic groups based on each query's topical focus. This means that users with similar tastes are categorized into the same group, and their search behaviour information is collected and used to train a topic category model. The structure of each topic category model is shown in Figure 3.2. Additionally we extract the representation for each topic category by computing the frequency of each document in the topic category and ranking all documents in descending order according to their frequency, then choosing the top few high frequency documents in the ranked list to generate a representation of this topic category.
- When a new user enters a query into the system: i) the query is passed to the standard IR component to retrieve a set of potential relevant search results from the available document collection; ii) the query is used to select a topic category for the current query by comparing the similarity between the query and the representations of each topic category.
- All data stored in this selected topic category is used by a recommender algorithm to output a prediction result list for the current query based on previous users' search behaviour in this category.

- The results of the IR search and RS predictions are then integrated using a linear combination of the scores for each retrieved document. This step means we use the prediction result list to re-rank the retrieval result list.

The details of the topic category and recommender components are described in the following subsections.

Generally, the aim of this proposed model is twofold. First, it addresses the short query problem for IR systems, when the user enters a short query to the system, and retrieval results cannot satisfy user information need. The recommender component can take advantage of other users' previous behaviour to give the prediction for this user. The recommendation results are intended to improve the IR results after the combination. Second, to ameliorate the key cold start problem of RSs. The Weighted Slope-One algorithm needs other users' search information to give predictions for the current user. If the topic category set S suffers from data sparsity, the prediction results will not be worth considering. In practice the data sparsity condition will apply for some user's queries. In this case there will not be an effective topical category available for users, and the output of the recommender algorithm will be empty or unreliable. In this case, retrieval results can be used to aid the recommendations to provide users with useful information. So this integrated IR model aims to exploit IR and RSs to benefit each other.

3.2 Recommender Component - Topic Category

The process of building topic categories is described in this section. Topic categories are built for recommender component, it includes all previous users' search activities, and for each current query, the best match topic category is selected and the information in the chosen category is used by the recommender component to generate the prediction results for the active query.

The topic category data set includes n different topic categories, where every topic category focuses on a topical area. Each topic category contains a number of similar interest user search history information (recorded search behaviour sets). Every user's search behaviour includes a list of documents and ratings that this user gave to each document. The structure of each topic category is shown in Figure 3.2.

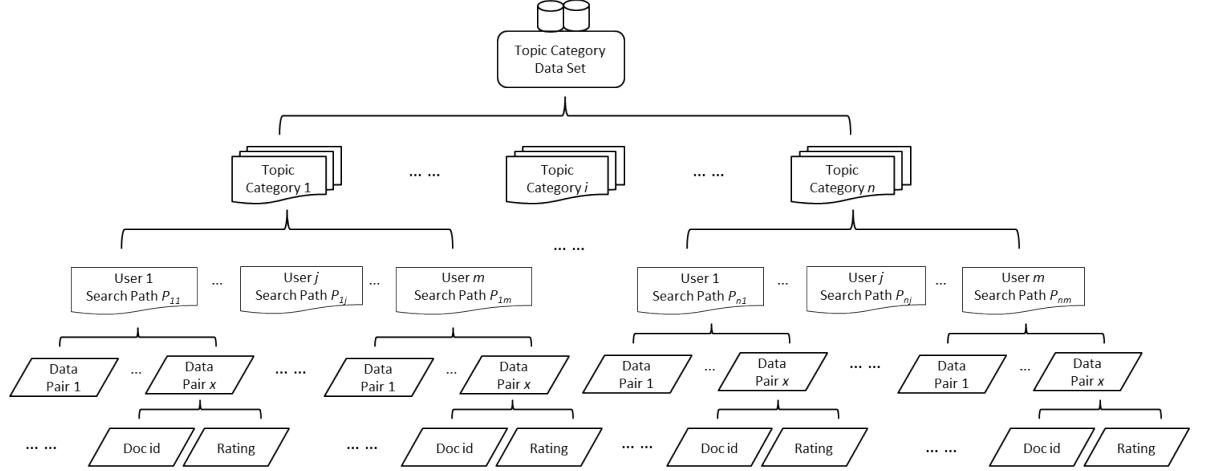


Figure 3.2 The structure of topic model

From Figure 3.2, we can see that each topic category i ($i \in [1, n]$) is a set $S = \{P_{1i}, P_{2i}, \dots, P_{mi}\}$, where m indicates the number of users' search information in the corresponding topic category, P_{im} is the search behaviour information of user m in topic category i , which contains m different similar interests users search behaviour information. This can be viewed as the following matrix specified in Equation (3-1). In this matrix, P_i ($i \in [1, n]$) is user i 's search path information, P_{ij} ($i \in [1, n], j \in [1, m]$), where n indicates the number of topic categories and m denotes the number of users search paths in each topic category. A user's search behaviour is a list of data pair (document, rating).

As mentioned, our research focuses on a user's query on a new topic in the condition that we cannot obtain any explicit information from this user with respect to this topic, so the only way

to provide the user with results customised based on previous search activities is to exploit other similar taste users search behaviour history. This topic category dataset is built to store all previous user search behaviour data. Since we have these users' search data, which contains the documents and ratings, a rating-based collaborative filtering (CF) algorithm is suitable for use in our system. The CF algorithm we selected for our initial experiment is the Weighted Slope-One algorithm (Lemire and Maclachlan, 2005) which was introduced in section 2.2.2.

$$S = \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ \dots \\ P_n \end{bmatrix} = \begin{bmatrix} P_{1,1} & P_{1,2} & P_{1,3} & \dots & P_{1,m} \\ P_{2,1} & P_{2,2} & P_{2,3} & \dots & P_{2,m} \\ P_{3,1} & P_{3,2} & P_{3,3} & \dots & P_{3,m} \\ \dots & \dots & \dots & \dots & \dots \\ P_{n,1} & P_{n,2} & P_{n,3} & \dots & P_{n,m} \end{bmatrix} \quad (3-1)$$

$P_{n,m}$ could be 0, because different topic category includes different number of users' search path information.

3.3 IR Component - Extended SMART Retrieval System

In this section, we introduce the SMART system, and the extended SMART retrieval system which we used for the IR component of the integrated model to generate the standard IR output in our experiment.

3.3.1 The SMART System

The SMART (System for the Mechanical Analysis and Retrieval of Text) system is an open source IR engine¹⁰ originally developed at Cornell University in the 1960s (Wikipedia¹¹). Many

¹⁰ SMART, "<ftp://ftp.cs.cornell.edu/pub/smart>"

¹¹ http://en.wikipedia.org/wiki/SMART_Information_Retrieval_System

important concepts in IR were developed as part of research on the SMART system, including the vector space model (VSM), relevance feedback and Rocchio classification. The processing steps are the same as described in section 2.1.2: tokenization, stop word removal, stemming, phrase extraction, document indexing and ranked document retrieval in response to an input query.

The SMART system automatically generates vectors for any given text collection and a set of queries and then uses the notion of vector similarity in computing the ranks of document vectors (Salton, 1971).

3.3.2 The Extended SMART System

In this chapter, we conduct our experiment on the extended SMART system. Ganguly (2008) extended the SMART retrieval system by implementing a standard language modelling approach into it. In the extended SMART system, the similarity score of an item j with respect to query q (denoted as $RS(j|q)$), is computed using the standard language modelling approach (see Equation (3-2)) (Ganguly, 2011). Equation (3-2) ranks an item j by the probability of this item generating the given query q . $P(t|j)$ represents the probability of generating a term t from item j and $P(t)$ is the probability of generating the term t from the whole data collection, λ being the smoothing parameter to account for the inverse document frequency (*idf*) factor of a term t .

$$RS(j | q) = \prod_{t \in q} \lambda \cdot P(t | j) + (1 - \lambda) \cdot P(t) \quad (3-2)$$

3.4 Linear Combination

In mathematics, a linear combination is an expression constructed from a set of terms by multiplying each term by a constant and adding the results (Wikipedia¹²). For example, a linear combination of x and y would be an expression of the form $ax + by$, where a and b are constant.

In this initial study, we combine the IR and recommender component using this simple linear combination of the lists of the outputs from the components. The combined score of item j (CW_j) is computed as shown in Equation (3-3).

$$CW_j = \alpha \cdot P_{(u)_j} + (1 - \alpha) \cdot RS(j | q) \quad (3-3)$$

where $P_{(u)_i}$ denotes the prediction rating that user u will give to item i . α is the combination parameter for prediction results.

3.5 Experimental Results

Since there is no experimental text collection including user search behaviour available for us to train topic categories for each topical domain, we used a simulation of user behaviour to conduct our initial experiments. We applied this approach to extend two existing IR test collections, the INEX 2009 ad hoc task and the TREC-8 ad hoc test collection. The two test collection experiments and the results obtained using this approach are described in the following subsections.

¹² http://en.wikipedia.org/wiki/Linear_combination

3.5.1 Initial Experiment on INEX 2009

This section describes our experiments on the INEX 2009 document collection. The INEX 2009 Wikipedia document collection comprises of 2,666,190 Wikipedia documents, and includes a total of 115 ad hoc search topics created by task participants. Since we exploit the previous user search behaviour in the integrated model, in order to conduct experiments on this collection, we needed to extend the INEX 2009 test collection by building simulated user search logs for it. The simulation of user search behaviour process is described in the following section.

3.5.1.1 Creation of Simulated Test Collection

For this investigation we simulated previous user search interaction information as follows. First, 20 topics were chosen from the INEX 2009 topic dataset. The criteria for choosing these 20 topics are that they should be 5 words or longer in length. 10 variations of each topic were created as a simulation of similar queries in the same topical area entered by previous users. Topic variants were made by randomly deleting one or two words from the original topic. Hence we needed to select topics statements of 5 or more words to maintain meaningful reduced queries. For example:

Original topic: *“Physicists scientists alchemists periodic table elements”*

Variations: 1) *Physicists scientists alchemists periodic results.*

2) *Physicists scientists alchemists table results.*

This is obviously a very simple strategy for creating topic variations, but enabled us to carry out our initial experiment. A particular issue which needs to be considered when modifying queries in this way is the potential impact on the set of documents which are relevant to each

topic. For this initial investigation, the relevance set was assumed to be the same as the original query for each topic variant.

- The 10 topic variants for each search topic were entered as queries into the extended SMART system to obtain 10 ranked lists of potentially relevant documents.
- The retrieved results for the 10 variations of each topic were categorized into one topic group. As described above, the aim of this step was to simulate results obtained for 10 different users interested in the same topic. The topic variations mean that slightly different result lists were obtained for each pseudo user search query.
- The 10 ranked lists for each topic variant were manually compared against the *qrel* file (relevance judgement) of the original topics to identify true relevant documents retrieved for each topic variant. Rating values were then assigned to each document randomly. The ratings simulated browsing time as an indication of implicit user feedback. These were assigned in the range 0.5-0.99 for each relevant document for the original topic, and 0.01-0.49 for documents which were non-relevant, based on the assumption that searchers will spend longer viewing the relevant documents.

Feedback from each of the top 150 documents (the value of 150 was chosen empirically) in the retrieved ranked lists with rating information was assumed to be a previous user's searching behaviour for this topic. The processed retrieved ranking lists for the 10 variants were integrated into one group and used as a topic category for their corresponding original query. We thus obtained our simulation data for previous users searching the document collection in each topical area.

3.5.1.2 Experiment Setup

Categorizing

Topic categories for the 20 selected topics were built as described in Section 3.5.1.1. For our experiment we assumed that a searcher had an interest in one of our 20 topical domains and entered the original search query to look for relevant documents that they might be interested in. The query was applied to the extended SMART retrieval system to obtain a ranked document list. This represents our baseline retrieval output.

The ranked IR retrieval list was then compared to the document pool of each topic category to identify the appropriate topic category for this query from those available. The topic category selection proceeded as follows: we assumed that the retrieved ranked list is a vector $Q = (d_{1,q}, d_{2,q}, d_{3,q}, \dots, d_{i,q})$, we have 20 topic categories (which means $n = 20$ in Figure 3.2) in our experiment $(R_1, R_2, R_3, \dots, R_{20})$. Topic category $k (k \in [1, 20])$ is a set S_k which contains $m (m \in [1, 10])$ of users' search behaviours, here $1 < m < 10$ means that we have 10 variants for each original topic to simulate 10 users search behaviour in each topic category, see Equation (3-1), and can also be viewed as a vector, *i.e.* topic category k can be seen as a vector $R_k = (P_{1,k}, P_{2,k}, P_{3,k}, \dots, P_{m,k}) (m \in [1, 10], k \in [1, 20])$, where $P_{i,j}$, is the result for one previous query in topic category k . The similarity between the query vector Q and each topic category vector R_k was calculated using Equation (3-4).

$$Sim(Q, R_k) = \sum_{j=1}^{20} f_{d_{j,q}, R_k} \quad (3-4)$$

where $f_{d_{j,q}, R_k}$ is the frequency of document d_j for the input query q in the topic category R_k . The topic category with the highest similarity was selected as the best matching topic category for the

input query. Here j ranges from 1 to 20, which means we only went through the top 20 documents in the retrieved ranked list obtained from SMART retrieval system for the input query (the modified training queries). This was then used to calculate the prediction of the rating that our current user would give to each of the available documents. Finally, the recommended ranking results were linearly combined with the baseline retrieval list to output our final integrated results. Figure 3.3 shows the combined results when the combination parameter α in Equation (3-3) is set to a different value. It can be observed clearly that we obtained the best combined output when the ratio of the recommender component contribution for combine results is 75%. So in this experiment, the parameter α was set to 0.25 according to the results shown in Figure 3.3.

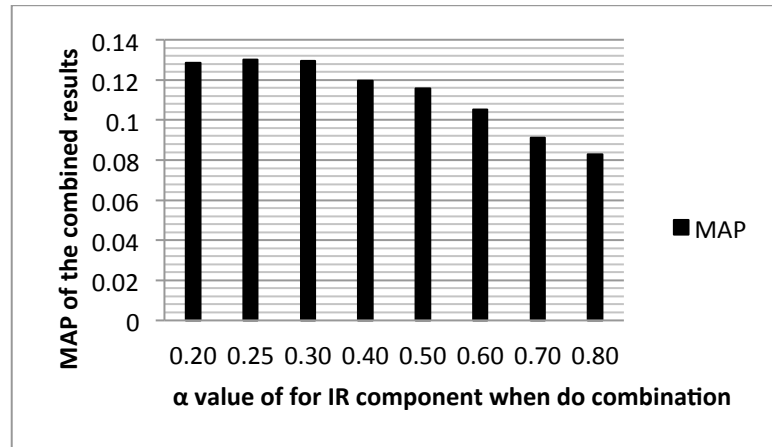


Figure 3.3 MAP of different α value when combine IR with RS

Compute Prediction

As described previously, we chose the rating-based Weighted Slope-One algorithm in the experiments introduced in this chapter, in order to compute the prediction, this algorithm needs the current user to rate a number of items. To satisfy this condition, similar to the pseudo-relevance feedback (PRF) method, we assumed that the top 2 documents (the top 2 documents in the initial retrieval ranked list for the current query) were relevant to the current user's information need. The obtained retrieval weights for these two documents were used as the

ratings that current user gave to these documents. The selected topic category data and the simulated current user ratings were used by the Weighted Slope-One algorithm to compute the prediction for the current user.

3.5.1.3 Evaluation Metrics

This section describes the evaluation metrics we used to evaluate the results we obtained: Mean Average Precision (MAP) and the precision at a cut-off position.

The standard scenario for use of MAP and rank cut-off precision in IR evaluation is to assume the presence of a collection of documents representative of a search task and a set of test topics (user queries) for the task along with associated manual relevance data for each topic. The relevance data for each topic is assumed to be a sufficient proportion of the documents from the collection that are actually relevant to that topic. “Sufficient” here relates to the fact that the actual number of relevant documents for each topic is unknown without manual assessment of the complete document collection for each topic, such assessment is impractical for all but trivial document collections. Several techniques are available for determining sufficient relevant documents for each topic (Tague et al., 1981; Buckley et al., 2006).

Precision

Precision is the fraction of the documents retrieved that are relevant to the user’s information need and is shown in Equation (3-5).

$$precision = \frac{|\{relevantdocuments\} \cap \{retrieveddocuments\}|}{|\{retrieveddocuments\}|} \quad (3-5)$$

Precision takes all retrieved documents into account. It can also be evaluated at a given cut-off rank which means only considering the top most results returned by the system. This measure is called precision at n ($P@n$) (Wikipedia¹³).

Recall

Recall is the fraction of the documents that are relevant to the query that are successfully retrieved.

$$recall = \frac{|\{relevant\ documents\} \cap \{retrieved\ documents\}|}{|\{relevant\ documnts\}|} \quad (3-6)$$

In binary classification, recall is often called sensitivity. So it can be looked at as the probability that a relevant document is retrieved by the query. It is trivial to achieve recall of 100% by returning all documents in response to any query. Therefore measurement of recall alone is not sufficient, but one needs to include measurement of the number of non-relevant documents also, for example by computing the precision¹³.

Average Precision

Precision and recall are single-value metrics based on the whole list of documents returned by the system. For systems that return a ranked list of documents, it is desirable to also consider the order in which the returned documents are presented. By computing a precision and recall at every position in the ranked sequence of documents, one can plot a precision-recall curve, plotting precision $p(r)$ as a function of recall r . Average precision computes the average value of $p(r)$ over the interval from $r = 0$ to $r = 1$, as shown Equation (3-7).

¹³ http://en.wikipedia.org/wiki/Information_retrieval

$$AveP = \int_0^1 p(r)dr \quad (3-7)$$

This is the area under the precision-recall curve. This integral is in practice replaced with a finite sum over every position in the ranked sequence of documents, shown in Equation (3-8).

$$AveP = \sum_{k=1}^n P(k)\Delta r(k) \quad (3-8)$$

where k is the rank in the sequence of retrieved documents, n is the number of retrieved documents, $P(k)$ is the precision at cut-off k in the list, and $\Delta r(k)$ is the change in recall from items $k - 1$ to k .

This finite sum is equivalent to the $AveP$ shown in Equation (3-9).

$$AveP = \frac{\sum_{k=1}^n (P(k) \times rel(k))}{\text{number of relevant documents}} \quad (3-9)$$

where $rel(k)$ is an indicator function equalling 1 if the item at rank k is a relevant document, zero otherwise. Note that the average is over all relevant documents and the relevant documents not retrieved get a precision score of zero¹³.

Mean Average Precision (MAP)

The mean average precision for a set of queries is the mean of the average precision scores for each query. While many evaluation metrics have been proposed for ad hoc type IR tasks, MAP is by far the most popular in general use (Baeza-Yates and Ribeiro-Neto, 2010). A detailed analysis of the behaviour of MAP is described in (Moffat and Zobel, 2008).

$$MAP = \frac{\sum_{q=1}^Q Ave P(q)}{Q} \quad (3-10)$$

where Q is the number of queries¹³.

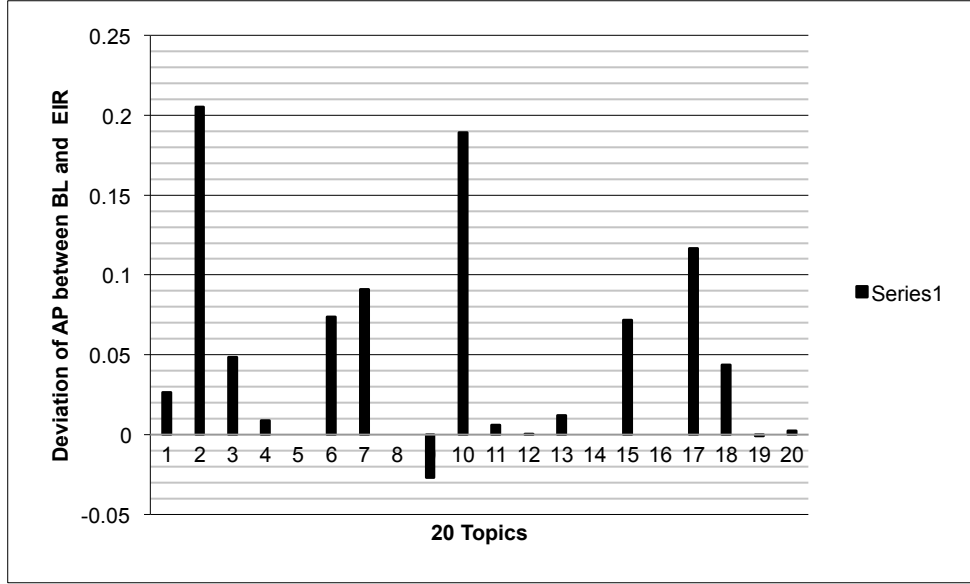
3.5.1.4 Experimental Results

Our aim here is to place the most relevant documents at the top of the list in order to give the user the most precise results which direct them to really interesting documents, i.e. we are more interested in precision at high rank cut-off of the retrieved list than recall of all relevant documents. Results for this experiment were calculated using the standard trec_eval¹⁴ software, and are shown in Table 3.1. Our baseline (BL) results are output by the SMART retrieval system. Our proposed integrated retrieval model is referred as the Enhanced IR model (EIR). Results are shown for precision at rank cut-off of 5, 10 and 20, and the standard MAP.

Topic	20_Topic_BL	20_Topic_EIR
MAP	0.0744	0.1303 (+75.03%)
P@5	0.2600	0.5000 (+92.30%)
P@10	0.2200	0.3560 (+61.81%)
P@20	0.1750	0.2000 (+14.23%)

Table 3.1 Retrieval results for 20 topics with simulated recommender training.

¹⁴ Trec_eval is the standard tool used by the TREC community for evaluating an ad hoc retrieval run, given the results file and a standard set of judged results. http://trec.nist.gov/trec_eval/



**Figure 3.4 Deviation of AP between BL and EIR approaches for 20 original test topics.
Calculated as $AP(EIR) - AP(BL)$**

From Table 3.1 we can see that the EIR approach achieves a MAP of 0.1303 which represents an impressive increase of +75.03% on the baseline results. The precision at top 5 cut-off ($P@5$) increases from 0.2600 to 0.5000 (+92.30%) compared to baseline. This partially matches our aim of seeking to promote relevant documents to the top of the ranked list. This demonstrates that the recommender algorithm has the potential to aid standard IR methods. Table 3.1 shows the deviation of average precision (AP) between the BL and EIR approaches (calculated by $AP(EIR) - AP(BL)$) for the 20 test topics. From Figure 3.4 we can clearly see that for the selected 20 topics, the average performance of our EIR model is better than the baseline. The reason that it does not perform well on all topics is that its results depend on the previous users visiting behaviour. If the topic category we choose for our current query is correct, and this topic category contains enough previous search information, we consider that useful predictions will be provided for the current query. For our 20 selected evaluation topics, 4 of the queries were assigned to the wrong topic category. Selecting the wrong topic category leads to unhelpful prediction results being provided to the users, with a decrease in the retrieval effectiveness.

Improving the reliability of topic category assignment is the focus of a further study in later chapters.

3.5.2 Experiment with the TREC-8 Ad hoc Task

Following this simple initial experiment, we revised our simulation approach to conduct a further experiment using an extended version of the standard TREC-8 ad hoc test collection.

3.5.2.1 Test Collection

The TREC-8 document collection is the set of documents on TREC disks 4 and 5 minus the Congressional Record documents. This collection contains 528,542 news articles which include material from the Financial Times Limited (1991, 1992, 1993, 1994), the Federal Register (1994), the Foreign Broadcast Information Service (1996) and the Los Angeles Times (1989, 1990). The 50 topics of the collection were again used to simulate previous users' search behaviour to build topic categories. Different from the previous experiment with the INEX collection, instead of randomly deleting terms from the original topics to make the variants, in this experiment, we automatically added or removed terms from the original query by using a text segmentation technique. The method used to generate the topic variants is based on an approach used to simulate topic reformulation within a searching session. The query variant generation process is described in the following subsection.

We made 20 variants for each of the original topics by using a text segmentation approach (Hearst, 1997). (Ganguly et al., 2011) observed that there are three patterns of query reformulations in real-life search behaviour: a) *specialization*, where the reformulated query expresses a more specialized information need as compared to the initial query; b) *generalization*, where the refined information need is more general and covers a broader scope in comparison to

the initial query; c) *drift*, where the reformulated query drifts away to another aspect of the initial information need instead of moving to more general or more specific needs. In effect, we assumed that the revised versions of the initial topic were topics used to state information needs closely related to that of the original topic, and were thus suitable for training our topic category models. Finally, the generated query variants were used as the users' information needs which were given to the search engine to output the retrieval results for the IR component in the integrated model.

3.5.2.2 Query Variant Generation Procedure

Text Segmentation (Hearst, 1997) is the process of decomposing a text into blocks of coherent textual content called segments, so that each segment's content is focused on one subtopic. Our generative model tried to utilize the fact that a term indicative of a more specific aspect of an initial information need is typically densely distributed in the textual contents of a segment whereas a more general term is distributed uniformly throughout the entire document text (Dang and Croft, 2010). In Choi (2000), it was found that the C99 segmentation algorithm is more accurate than other existing text segmentation algorithms. So we chose the C99 algorithm for text segmentation in this work. C99 uses a matrix-based ranking and a clustering approach in order to relate the most similar textual units.

Terms are scored based on the following two factors:

- How frequently a term t occurs in a segment S , denoted by $tf(t, s)$, and how exclusive the occurrence of t in S is as compared to other segments of the same document, denoted by $|S|/sf(t)$, where $|S|$ is the number of segments in this document and $sf(t)$ is the number of segments in which t occurs.

- How rare the term is in the entire collection, measured by the document frequency (df), the assumption being that rare terms are more likely to be specific terms.

For more specific reformulations of the original query, we computed the term scores for the most similar segment to the query, as shown in Equation (3-11). Adding terms from this segment can potentially shift the original query to a more specific information need.

$$spec(t, s) = \alpha \cdot tf(t, s) \frac{|S|}{sf(t)} + (1 - \alpha) \cdot \log \frac{|D|}{df(t)} \quad (3-11)$$

Equation (3-11) assigns higher values to terms which occur frequently in a segment, occur only in a few segments, and occur infrequently in the collection. The working steps of the method to generate more specific query variants are as follows:

- 1 For each top ranked R documents for query q do steps 2-5.
- 2 Segment a document d into $\{s_1, s_2, \dots, s_n\}$ by executing C99.
- 3 Let s_{sel} be the segment with the maximum number of matching query terms for query q specific reformulations.
- 4 Score each term $t \in s_{sel}$ by $spec(t, s_{sel})$ by Equation (3-11) for specific reformulation.
- 5 Average the term scores over documents.
- 6 Sort each term by its score and add (or substitute) the top n new terms to the original query for more specific reformulation.

In contrast to a more specific term, a more general term is distributed uniformly throughout the entire document text. So an obvious choice for generating more general variants is to score a term based on the combination of term frequency in the whole document (instead of frequency in

individual segments) and segment frequency (instead of inverse segment frequency) where $tf(t, d)$ is the number of occurrences of t in d as shown in Equation (3-12).

$$gen(t, d) = \alpha \cdot tf(t, d) \frac{sf(t)}{|S|} + (1 - \alpha) \cdot \log \frac{|D|}{df(t)} \quad (3-12)$$

Similar to the process of generating more specific query variants, the working steps of the method to generate more general query variants are as follows:

- 1 For each of the top ranked R documents for query q do steps 2-5.
- 2 Segment a document d into $\{s_1, s_2, \dots, s_n\}$ by executing C99.
- 3 Score each original query term t by Equation (3-12).
- 4 Average the term scores over documents.
- 5 Retain the top n terms in the query removing the rest for a general reformulation.

Based on (Ganguly et al., 2011), the same parameters were used for query variant generation in this study. For the more general reformulations, we used the description field of the TREC-8 topics as the initial query. For more specific reformulations, we used the title fields as the initial query. Figure 3.5 shows an example of the format of TREC-8 topic.

Five top ranked documents for reformulating each query with $\alpha = 0.5$ were used to compute the $spec(t, s)$ and $gen(t, d)$ scores. To generate more specific variants we added at most 3 additional terms with high $spec$ scores to the original query, and for generating more general variants we removed the low scoring gen terms and retained at most 2 terms from the description part of the original TREC topics. These numbers were chosen based on the experiments Ganguly conducted in (Ganguly et al., 2011).

TOPIC 51
<pre> <top> <head> Tipster Topic Description <num> Number: 051 <dom> Domain: International Economics <title> Topic: Airbus Subsidies <desc> Description: Document will discuss government assistance to Airbus Industrie, or mention a trade dispute between Airbus and a U.S. aircraft producer over the issue of subsidies. <smry> Summary: Document will discuss government assistance to Airbus Industrie, or mention a trade dispute between Airbus and a U.S. aircraft producer over the issue of subsidies. <narr> Narrative: A relevant document will cite or discuss assistance to Airbus Industrie by the French, German, British or Spanish government(s), or will discuss a trade dispute between Airbus or the European governments and a U.S. aircraft producer, most likely Boeing Co. or McDonnell Douglas Corp., or the U.S. government, over federal subsidies to Airbus. <con> Concept(s): 1. Airbus Industrie 2. European aircraft consortium, Messerschmitt-Boelkow-Blohm GmbH, British Aerospace PLC, Aerospatiale, Construcciones Aeronauticas S.A. 3. federal subsidies, government assistance, aid, loan, financing 4. trade dispute, trade controversy, trade tension 5. General Agreement on Tariffs and Trade (GATT) aircraft code 6. Trade Policy Review Group (TPRG) 7. complaint, objection 8. retaliation, anti-dumping duty petition, countervailing duty petition, sanctions <def> Definition(s): ... </pre>

Figure 3.5 an example of the format of TREC-8 topic.

Generating simulated user search history

To generate more specific queries, we started by retrieving 30 documents for each original TREC-8 topic by using its title field. We then divided the 30 documents into 10 groups: 3 consecutive documents were grouped such that the top 1-3 documents form a group, the top 4-6 documents formed another group and so forth. The three documents in each group were used to generate one query variant as described above. For reformulations to more general queries the process was similar, the only difference was that we used the description part of the TREC-8 topic as the initial query to facilitate removal of terms to form a more general variant topic. Titles were not used as initial queries for general variants because titles were too short and not conducive to removal of terms.

- We thus generated 20 query variants (10 more specific and 10 more general queries) for each original TREC-8 topic.

- Each query variant was seen as a user's query. This query was passed to the search engine to obtain the retrieval results.
- The retrieval ranking was used as this user's activity data.
- Finally, the additional 20 user's activity data were used to build a topic category model for each topical domain.

Since the TREC-8 test collection has the same problem as the INEX 2009 collection, that is that there is no users' search behaviour data available for the recommender component to compute the prediction results, we also need to simulate the users' click-through information. The simulated click-through data for each user was built as follows:

- The extended SMART retrieval system was used to obtain an initial ranked list of documents for each topic variant.
- Both variant queries and their obtained resulting retrieval information were used to simulate users' visiting behaviours to build topic categories. The process for building the topic categories was similar to that used in the INEX experiment which used the top ranked document in the retrieval list and their weighting score as the user logs.

In order to make the simulated scenario more realistic, the simulated users' click-through data were built for two conditions:

- Perfect condition: using a relevance assessment file to build topic categories, this process is shown in Figure 3.6.
- Poor condition: since the perfect condition is an ideal one, it never occurs in the real world, thus, we attempted to examine the effectiveness of our integrated model in a

poorer setting. This condition used pseudo-relevance feedback (PRF) to build topic categories, this process is shown in Figure 3.7.

We regard this as a realistic experiment, since in a real world setting, it is not realistic for the user to give accurate feedback on all occasions. We expect that feedback will contain some noisy non-relevant documents. Thus based on the perfect and poor condition, we also built several different styles of topic categories for our experiments, such as:

- Add noisy documents to the perfect condition.
- Randomly assigning different types of ratings to documents
- Discarding documents with marginal relevance ratings or intermediate ratings from the retrieval list, because we assume that documents with extreme values or with values in the middle range of $[0, 1]$ are not helpful for deciding their relevance status.

We explored all these methods to simulate real world users for our experiment investigating the impact of noisy data on our retrieval method. The procedures for building these topic categories are described in the next paragraph.

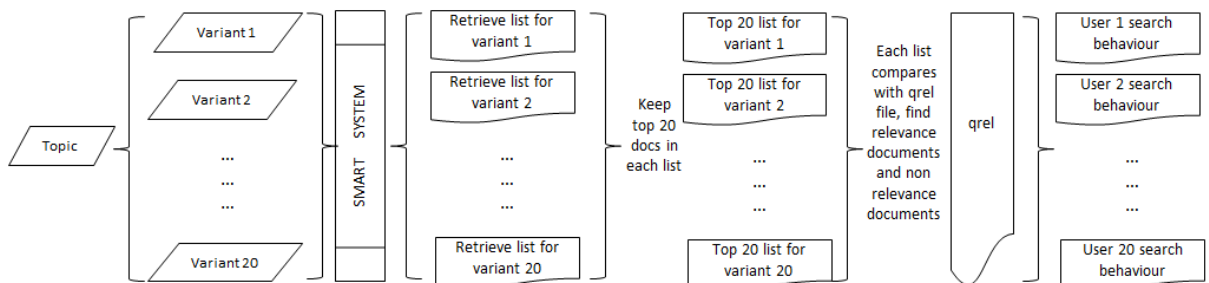


Figure 3.6 The process of building one topic category for the perfect condition

Four variants for topic categories were used for the perfect condition recommendation:

- Binary rating values were set for documents which involve using either 1 or 0 (assigned to top 20 documents in the initial retrieval list, 1 for relevant documents and 0 for non-relevant documents) as the visiting time of documents (EIR_B).
- Involving real numbers in the range $[0, 1]$ taken as the viewing time (EIR_N).
- Similar to EIR_N, but discarding the documents with viewing times in a specific range (EIR_ND), because we assumed that a document rating in this specific range is not a clear indicator for deciding the relevance of this document.
- Similar to EIR_N, but with n noisy documents in each user's browsing information. This meant that n non-relevant documents with higher rating were inserted into each user's browsing list (EIR_NN(n)). We believe EIR_NN(n) is a realistic condition to explore since users frequently spend time on browsing non-relevant documents.

These four topic category variants were built to examine the effectiveness of the integrated model. The building procedure was as follows:

- 1) For each original query (each query variant can be seen as a user), we made 20 variants, 10 more specific variants and 10 more general ones.
- 2) We used the extended SMART retrieval system to retrieve an initial ranked list for each of these 20 topic variants.
- 3) We assumed that each user goes through the top ranked 20 documents in the initial ranking, thus we kept top 20 documents from each retrieval list to simulate each user's search activity.
- 4) Each of the top 20 documents in the list was compared with the qrel file for this topic (TREC-8 relevance assessment file) to mark the relevant and non-relevant documents.

5) The corresponding viewing time information was assigned to each document based on the relevance assessment for each of the following conditions:

- EIR_B: binary rating value by assigning 1 for relevant documents and 0 for non-relevant documents.
- EIR_N: true relevant documents are assigned a random number in the range of $[0.5, 0.99]$; a random number in the range of $[0.1, 0.49]$ was assigned to non-relevant documents.
- EIR_ND: the same browsing time information was assigned for EIR_N, but after this, we discarded the mid-range viewing times in the range $[0.3, 0.6]$. The purpose of this method was that we assumed that users always prefer to provide really good or bad ratings. If the document meets their information need, users give a good rating; if not, a bad rating will be given to the document. Thus to achieve this goal, we discarded documents with ratings in the mid-range.
- EIR_NN(n): the same browsing time information was used as for EIR_N, n noisy documents were inserted into each user's browsing list. A noisy document was a non-relevant document with high rating which may exist in a previous user's browsing list. Since we simulated 20 users' search activities in each topic category, $20 \times n$ noisy documents were inserted into each topic category in total.
- EIR_NND(n): based on EIR_NN(n), involving discarding documents with viewing times in the mid-range between $[0.3, 0.6]$.

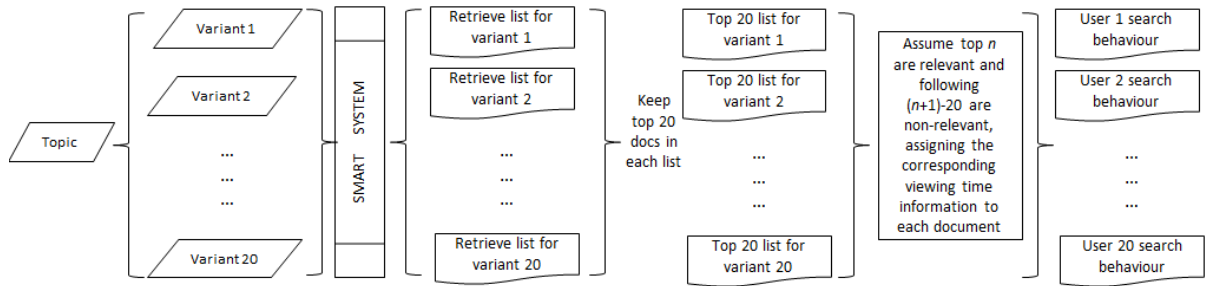


Figure 3.7 The process of building one topic model for the poor condition

For the poor condition, which used a method similar to the pseudo-relevance feedback (PRF) method in the IR system, the process of building topic categories was similar to the perfect condition, only the last few steps were changed. The first 3 steps were same as the perfect condition. The difference in this process was from step 4, which instead of using relevance assessment to identify the relevant and non-relevance document of the top 20 documents in the initial ranking, assumed that top n documents were relevant and that rest were non-relevant.

- 1) For each original query (each query variant can be seen as a user), we made 20 variants, 10 more specific variants and 10 more general.
- 2) The extended SMART retrieval system was used to retrieve an initial ranked list for each of the 20 variants.
- 3) We again assumed that each user only goes through the top ranked 20 documents in the initial ranking, so we kept only the top 20 documents from each list.
- 4) For the top 20 document in the retrieved list, similar to PRF, we assumed that the top $n(n < 20)$ documents were relevant, and that the rest of the $20 - n$ documents (from $n + 1$ to 20) in the ranked list were non-relevant.
- 5) The corresponding viewing time information was assigned to each document based on their relevance assessment.

- EIR_PRF(n): In the PRF based poor condition, the top n documents were assigned a random number in the range of $[0.5, 0.99]$ and documents whose rank between $(n + 1)$ to 20 were assigned a random number in the range of $[0.1, 0.49]$.
- EIR_PRFD(n): based on EIR_PRF(n), but in order to discard the documents with ambiguous relevance rating, this run involved discarding documents with viewing times in the mid-range $[0.3, 0.6]$.

3.5.2.3 Experiments and Results

The different types of topic categories were used to conduct the following set of experiments. We assumed that each user was interested in one of the 50 original topics in the TREC-8 dataset.

Categorizing the Query to Best Match Topic Category

This section describes the procedure for categorizing the current query into the best matched topic category. Since there were over 300 documents included in each topic category, the document focus on the topic may vary. It was a challenge to find the best matching topic category for the current query. Arora et al., (2008) show that summarizing the documents and query to a shorter length is a good approach to improve the retrieval effectiveness. In this experiment, we chose to build a representation for each topic category by summarizing multiple documents into a short one. MEAD (Radev et al., 2003) is a widely used platform for document summarization and evaluation. Since it suited our purpose, we used MEAD to generate the representation for each topic category. We next give a brief overview of the functionality of the MEAD platform.

The MEAD Platform

MEAD¹⁵ is the most sophisticated publically available platform for multi-lingual summarization and evaluation (Radev et al., 2003). The platform implements multiple summarization algorithms such as position-based, centroid-based, largest common subsequence, and keywords. We use the centroid-based multi-document summarizer to generate a centroid document for each topic category, and use this as the representation for the topic category. The centroid is a set of words that are statistically important to a cluster of documents.

To generate the summarized document for a set of documents, MEAD decides which sentences to include in the extract by ranking them according to a set of parameters. The input to MEAD is a set of documents. These documents are first segmented into sentences. The size of the final summary is determined by the compression rate R . The output is a sequence of $n * R$ sentences from the original documents presented. For example, if the set contains a total of 50 sentences ($n = 50$) and the value of R is 20%, which is a standard rate, the output of MEAD will contain 10 sentences. Since we chose to use a centroid-based multi-document summarizer to generate a centroid document for each topic category, the centroid value was used to compute the salience of a sentence in the document set. The centroid value C_i for sentence S_i is computed as the sum of the centroid values $C_{w,i}$ of all words in the sentences, shown in Equation (3-13).

$$C_i = \sum_w C_{w,i} \quad (3-13)$$

¹⁵ <http://www.summarization.com/mead/>

Generating a Representation for a Topic Category

As mentioned above, we use the query variants of the user's search query. We again simulated the user's search activity by taking the top 150 documents from the retrieval list for this query and assigning a rating in the range [0, 1] to the list of documents randomly. Finally, these 150 documents with their rating was used as the user's search behaviour. 20 variants were made for each of the original TREC-8 topics. Each variant was used to generate a user's search behaviour. The 20 simulated user search behaviours were used to build a topic category. After building the topic categories, the 5 most frequent viewed documents in each topic category were used to generate a summary using the MEAD system with R set to 10%, which was then used as the representation for the corresponding topic category.

Categorizing

When a user enters a query on a new topic for them, the cosine similarity was used to compute the similarity between this query Q and the representation of each topic category (C_i) and the most similar one for this query was selected, as shown in Equation (3-14).

$$similarity(Q, C_i) = sim_{cosine}(Q, C_i) \quad (3-14)$$

Computing Predictions

The rating-based Weighted Slope-One recommender algorithm was again used to compute item prediction. Besides the topic category data, the Weighted Slope-One scheme also needs the current user's search behaviour to compute the prediction. So similar to the INEX test collection experiment, we also assumed that the top 2 documents of the initial ranking for the user's query were relevant to this query. The difference from the first experiment was the ratings given to these top 2 documents. Since we assumed they are relevant to the current user's query, we assigned a high rating to them, so we assigned a viewing time value randomly in the range of

[0.5, 0.99] to these 2 documents, and used them as the current user's search information. Both the selected topic category data and the current user's search data were given to the recommender algorithm to output the prediction document list for this user.

Combination

We again used a linear combination of the initial ranked list of the user query and the prediction ranking to re-rank the initial ranking, as shown in Equation (3-3) In this experiment, the combination parameter was again set to $\alpha = 0.25$.

Results

The initial retrieval results obtained from the extended SMART system for the TREC-8 topics were used as our baseline (BL) for the comparison with our approach. For our experiment, we performed 7 runs on these two different types of topic categories (in perfect and poor conditions):

- 1 EIR_B: binary rating value by assigning 1 to true relevant documents and 0 for non-relevant documents. (run1)
- 2 EIR_N: true relevant documents assigned a random number in the range of [0.5, 0.99]; a random number in the range of [0.1, 0.49] was assigned to non-relevant documents. (run2)
- 3 EIR_ND: the same browsing time information was assigned as EIR_N, but discarding the mid-range viewing times in the range [0.3, 0.6], because the viewing time in this range does not reflect the relevance of documents explicitly. (run3)
- 4 EIR_NN(2): the same browsing time information same as EIR_N, 2 noisy documents were then inserted into each user's browsing list. (run4)
- 5 EIR_NND(2): based on EIR_NN(2), involving discarding documents with viewing times in the mid-range between [0.3, 0.6]. (run5)

- 6 EIR_PRFD(10): based poor condition, top 10 documents were assigned a random number in the range of [0.5, 0.99] and documents whose rank between 11 to 20 were assigned a random number in the range of [0.1, 0.49]. (run6)
- 7 EIR_PRFD(10): based on EIR_PRFD(10), involved discarding documents with viewing times in the mid-range [0.3, 0.6]. (run7)

	BL	run1	run2	run3	run4	run5	run6	run7
MAP	0.2409	0.3693 (+53.3%)	0.3425 (+42.17%)	0.3427 (+42.26%)	0.2686 (+14.50%)	0.2784 (+15.57%)	0.2409 (+0%)	0.2410 (+0.04%)
P@5	0.4120	0.8240 (+100%)	0.8120 (+97.09%)	0.8120 (+97.09%)	0.5520 (+33.98%)	0.6010 (+45.87%)	0.4200 (+1.94%)	0.4200 (+1.94%)

Table 3.2 The results of 7 runs compare to baseline

Table 3.2 shows that all runs using the perfect condition recommendation increase MAP significantly compared to the baseline. The MAP of the binary run (EIR_B) increases from 0.2409 (baseline) to 0.3693 (+53.3%), and P@5 increases from 0.4120 (baseline) to 0.8240 (+100%). Compared to this ideal scenario, the more realistic run EIR_N performs a little worse, but still has a big improvement with respect to the baseline, as can be seen from an increase in MAP by 42.28% over the baseline. Its P@5 also significantly improves to 0.8120.

An interesting observation is the lack of significant improvement in MAP in the presence of noisy non-relevant documents in the recommender. It can be seen that for EIR_NN(2), the MAP only increases by around 11.5% with respect to the baseline, which means that this approach is sensitive to noisy documents.

We conducted additional experiments by selecting documents with viewing times at the two extremes of high values (>0.6) and low values (<0.3), presuming that extreme values can be useful as strong indications of the relevance or non-relevance of a document. The documents

with viewing times in the mid-range interval, i.e. $[0.3, 0.6]$, were discarded. We found that variation in the cut-off ranges had little effect on the retrieval effectiveness, and we finally chose $[0.3, 0.6]$, since it resulted in the largest improvement in retrieval. This approach of using the extreme viewing times for recommendation does not much affect the results of the perfect scenario EIR_N.

From our results shown in Table 3.3, it can also be seen that poor condition is not an attractive choice for recommenders because the results do not improve compared to the baseline, either with all documents used (run6) or with documents selected in the two extreme ranges of viewing times (run7). A possible reason for this could be the assumption of pseudo relevance of results in the input of a number of non-relevant documents to the recommender, amounting to degradation in performance due to sensitivity of recommenders to noisy information.

3.6 Summary

This chapter has presented the initial experiments we carried out to examine our proposed integrated IR model which focuses on combining IR with a recommender algorithm. We took advantage of recommender techniques to aid a standard IR system, with the aim of improving the retrieval effectiveness by utilizing previous user search behaviour data. In these experiments, a rating-based Weighted Slope-One recommender algorithm was chosen to compute the prediction ranking for recommender components in two datasets based on the simulated previous users search data. The results show:

1. Positive results were obtained in both sets of experiments. These provide evidence in support of a positive answer to our first research question: we showed that

recommender algorithms can help the IR model to obtain better retrieval results in our initial simulated settings.

2. The experiment conducted on the TREC-8 ad hoc test collection showed that noisy documents affect results significantly, which means that this method is noise sensitive. Thus, the existence of noisy documents in previous users' search logs has the potential to impact dramatically on the effectiveness of this method.
3. In these initial experiments, we exploit a rating-based recommender algorithm in the integrated model to output prediction ranking. This method depends on the ratings from previous users to compute recommendations which may fail to handle the noisy problem in our approach. Because sometimes users may give noisy documents high ratings. In this condition, the rating-based algorithm cannot filter noisy documents based only on ratings. This noise sensitivity problem is further investigated in later chapters

In this chapter, we examined the effectiveness of using the rating-based Weighted Slope-One recommender algorithm in the IR model, and obtained positive results. In the next chapter, we investigate the effectiveness of other existing recommender algorithms in the IR model. The weakness of these algorithms in the model is observed and a simple improvement is made to the Weighted Slope-One algorithm to make it more suitable for the integrated IR model.

Chapter 4

Investigating the Effectiveness of Alternative Recommender Algorithms and Fusion Methods in the Integrated Information Retrieval Model

This chapter reports our investigation of the application of alternative recommender algorithms into our integrated model and their performance. In Chapter 3, we described our initial experiments examining our proposed integrated retrieval model incorporating the Rating-Based Weighted Slope-One algorithm to compute the recommendation for the current user. While this initial study achieved positive results, it is important to consider whether other recommender algorithms may be more effective. In this chapter, we compare the performance of alternative recommender techniques in our integrated information retrieval (IR) model. These experiments examine both content-based filtering and collaborative filtering. As will be shown, our experimental results reveal a weakness of the current recommender algorithms when applied for enhancing IR. We present a simple improvement to the rating-based collaborative filtering (CF) algorithm which overcomes this weakness. Despite the simplicity of our modified algorithm, we demonstrate that it obtains noticeably better results than existing recommender techniques. We also examine alternative fusion methods for the combination of the IR and recommender system

(RS) components in the last step of the integrated IR model. Also in contrast to Chapter 3, instead of using simulated users' query and search behaviour to examine the effectiveness of the integrated model, which may not be convincing because real world users are more unpredictable, in this chapter all experiments are conducted using data provided by human subjects. This data was collected as part of an extended version of the FIRE 2011 personal information retrieval (PIR) test collection (Ganguly et al., 2011a). The FIRE 2011 English ad hoc document collection is composed of news articles from the Indian newspaper *The Telegraph* from 2001 to 2010 and news from Bangladesh. We extend this document collection by extraction of topics from the collection, and asking volunteer users to contribute queries for each topic, recording their search activities which we use to build topic categories, and asking users to provide relevance assessments for documents returned in response to their query.

The structure of this chapter is as follows: first we introduce the process of collecting information from users to build the test collection for our experiments; in the second part we describe the fusion methods we investigated in our experiments; then the experimental procedure is introduced and the results obtained described; subsequent analysis of these results is shown to reveal the weakness of existing RS methods for our task and a novel RS method is then proposed and evaluated.; finally conclusions of the chapter are summarized.

4.1 Test Collection Building¹⁶

¹⁶For the FIRE 2011 data collection, this user behaviour collection website was built by Debasis Ganguly, he extracted 15 topics from the dataset. The current study expanded this to 27 topics, and volunteer users were asked to contribute more queries than were provided in the original FIRE 2011 collection. Analysis of the data, such as extraction of dwell time for each document and remainder of the data collection work were all carried out as part of this study by the author.

This section introduces the test collection used in the experiments conducted in the remainder of the thesis. In Chapter 3, we simulated users' search behaviour to build topic categories for test collections to conduct our experiments to examine the effectiveness of our integrated IR model. Instead of using simulation to build test collections, from this chapter onwards, our experiments are conducted on a test collection built by collecting topic categories, user queries, user search activities and user ratings from real users. (Liu et al., 2008) demonstrates that the length of time which a user stays on a document is a good indicator of the quality and importance of the document. Thus we extracted user ratings based on the dwell time that users spend on the corresponding items. The process of building the test collection for our experiments is described in the following subsections.

4.1.1 Topic Extraction

In order to investigate the effectiveness of the methods introduced in the previous section, the user behaviour data collected for the FIRE 2011 PIR task (Ganguly et al., 2011a) was used for the evaluation. This dataset is based on the FIRE 2011 English ad hoc document collection composed of news articles from the Indian newspaper *The Telegraph* from 2001 to 2010 and news from Bangladesh, comprising almost 400k documents in total. This dataset contains user search log information collected from a number of volunteer users. It is an ideal dataset to explore our proposed method to utilize previous users' search information to compute the relevance of each document to improve the IR results. We identified 27 topic areas from the dataset content. Table 4.1 shows examples of the topic categories. The full set of 27 topic categories is listed in Appendix A.

Topic Categories
<i>Indian Tourism</i>
<i>Social impact on land acquisition</i>
<i>Relation of India with its neighboring countries</i>
<i>Indian paintings and painters</i>
.....

Table 4.1 Example of Topic Categories Extracted from Dataset

4.1.2 Collect User Behaviour Data

In the FIRE 2011 PIR task, participants are academic students. The following steps were carried out to collect users' search behaviour information. The instructions given to the participants are shown in Appendix B.

- A number of participants volunteered to search the document collection in one of the 27 provided topic areas. Each participant selected one of the 27 topics themselves to ensure that this was an area in which they were knowledgeable and interested. They then created a topic statement (query) related to the chosen topic. Table 4.2 shows an example of one topic and the queries collected from several users.

Topic Category	Query
<i>Indian Tourism</i>	<i>Place to visit in Indian Sightseeing in Indian Tourism in Indian Touristic sights in Indian</i>

Table 4.2 Example of Topic Categories and User Entered Queries

- The participant then submitted their query to the Terrier retrieval system which returned a ranked list of potentially relevant news documents to the participant. They then began viewing documents from the list returned in response to their query.

They were able to click and view any documents in which they were interested based on the displayed document snippets. They could repeat this action until they found the information they needed or gave up. The participant's activities were tracked and logged. The log recorded information including the participant's username, the topic category selected, the contributed query, the returned documents viewed, and the dwell time that they spent on each document. Table 4.3 shows examples of recorded user behaviour data in each topic category. The time includes the date and the exact time that the user opened the corresponding document. This time stamp information is used to extract the dwell time that user spent on each document. The dwell time is used in our experiments as the rating that the user gave to each document.

	Topic	query	docID	Time
Test1	Indian tourism	Places to visit in Indian	1050304_nation_story_4450668.utf8	2011-08-23 10:33:06
			1040715_foreign_story_3498066.utf8	2011-08-23 10:34:20
			1040715_foreign_story_3498066.utf8	2011-08-23 10:36:02
		
Test 2	Indian tourism	Sightseeing in Indian	1100105_nation_story_11944294.utf8	2011-08-25 14:45:42
			1100130_nation_story_12045663.utf8	2011-08-25 14:52:43
			1100505_nation_story_12413798.utf8	2011-08-25 14:52:32
			1050304_nation_story_4450668.utf8	2011-08-25 14:53:43
			1080317_nation_story_9028502.utf8	2011-08-25 14:56:06
		

Table 4.3 Example of recorded users search behaviour

- In addition, each participant was also required to provide relevance assessments for the queries they provided. They were asked to read the top 30 documents in the initial ranked list returned for each of the queries that they entered, and to mark relevant documents which addressed their information need. These selected relevant documents were used as the relevance assessment data for our experiments. Note

that this relevance assessment collection process was separate from the search log collection procedure.

In total, 26 participants contributed 150 queries for the 27 topics. It should be noted that since the participants were given free choice of topic, the queries are distributed unevenly over the available topics. One query was randomly selected from each topic category to be used as test query for this topic. This resulted in 123 queries to be used as a training set and 27 queries, one for each topic category selected from the collected queries randomly, to be used as the test topic set. Appendix A shows the 27 topic categories and queries in each category.

From the participants search information shown in Table 4.3., “User ID” is user’s ID information, “Topic” indicates the topic area users selected, “query” is the query they submitted in the selected topic area, “docID” denotes the documents this user viewed and “Time” is the time that each document is opened by this user. The user log information was processed to extract dwell time information on each document.

4.1.3 Dwell time extraction

We segmented the viewing session based on the query. If the query changed, a new session was started. For every viewing session, we used the difference between the time of the second document and that of the first document as the observed dwell time on the first document. For example, in Table 4.3, for searcher test1’s search log, the view time for document ‘1050304_nation_story_4450668.utf8’ was calculated as $1034.20 - 1033.06 = 1.14$. For the last document in a session, we used the following heuristic to decide its observed staying time; we computed the average viewing time from the distribution of observed viewing time of documents in all the records of this user and took this as the observed viewing time for the document.

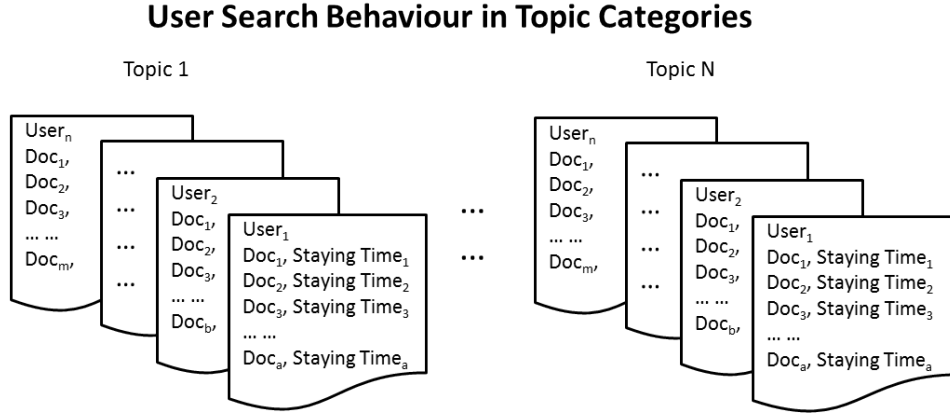


Figure 4.1 Structure of user logs in N different topic categories

After extracting the dwell time information, we could build a user search log for the query, and cluster the user log into relevant topic categories based on the “Topic” they chose. For each topic category, we recorded each user’s search behaviour. Figure 4.1 shows the previous user logs data and the structure of user logs in N different topic categories.

4.2 Fusion Methods

In the initial experiments we described in the previous chapter, we simply used a linear combination method to combine the outputs of the IR and recommender components in the integrated model. In this linear combination, we need to adjust the parameters for the two components to achieve the best results. However, in a real world setting, it is unrealistic to be able to tune parameters for different cases. Thus, in the experiments conducted in this chapter, we examine the use of several alternative fusion operators which do not contain adjustable parameters, with the objective of replacing the linear combination method to merge outputs from two components. The fusion operators we explore were introduced in Shaw and Fox (1994) and can be summarized as follows:

CombSUM

The CombSUM operator forms a combined document score which is simply the sum of the document scores achieved by the different schemes being combined as shown in Equation (4-1).

$$CombSUM_i = \sum_{i \in (IR\ scheme \cup RS\ scheme)} score_i = score_{IR_i} + score_{RS_i} \quad (4-1)$$

where document i is the document associated with scores from $IR\ scheme \cup RS\ scheme$, and $IR\ scheme$ denotes the IR ranked list and $RS\ scheme$ is the prediction ranked list.

CombMAX and CombMIN

The CombMAX and CombMIN operators indicate the max and min score of the document scores by the alternative ranking schemes, as shown in Equation (4-2) and Equation (4-3).

$$CombMAX_i = \max(score_{IR_i}, score_{RS_i}) \ (i \in (IR\ scheme \cup RS\ scheme)) \quad (4-2)$$

$$CombMIN_i = \min(score_{IR_i}, score_{RS_i}) \ (i \in (IR\ scheme \cup RS\ scheme)) \quad (4-3)$$

CombANZ and CombNBZ

$CombANZ_i$ is specified as the same sum of document scores as $CombSUM_i$ but divided by the number of ranking schemes which contain document i . $CombNBZ$ is defined as $CombSUM_i$ multiplied by the number of ranking schemes which contain this given document. $CombANZ$ and $CombNBZ$ are shown in Equation (4-4) and Equation (4-5) respectively.

$$CombANZ_i = \frac{CombSUM_i}{number\ of\ nonzero\ score_i} \quad (4-4)$$

$$CombNBZ_i = CombSUM_i \cdot number\ of\ nonzero\ score_i \quad (4-5)$$

CombNorm

Lee (1995) proposed a normalization method which utilizes the maximum and minimum scores for each weighting scheme as shown in Equation (4-6). The summation of these normalized scores for document i is shown in Equation (4-7)

$$NormalizedScore_i = \frac{score_i - Minscore}{Maxscore - Minscore} \quad (4-6)$$

$$CombNorm_i = \sum_{i \in (IR\ scheme \cup RS\ scheme)} NormalizedScore_i \quad (4-7)$$

4.3 Experimental Setup

Based on the extended FIRE 2011 test collection introduced in Section 4.1, this section describes the procedure for conducting our experiments which investigated exploiting various recommender algorithms in the integrated model.

4.3.1 IR Component

In contrast to the initial experiments in Chapter 3 which use the SMART language model retrieval model to obtain the retrieval results for IR component, in this chapter the Terrier BM25 retrieval model was used to generate ranked lists for the IR component. The BM25 probabilistic model was defined in (Robertson et al., 1995), and is based on a body of prior work on probabilistic ranking. The BM25 ranking model is shown in Equation (4-8). A Terrier system stopword list of 733 words was used with a Porter stemmer (Porter, 1980) to pre-process the input text. A standard TREC formatted ranked list of 1000 documents was returned for each query.

$$score(D, Q) = \sum_{i=1}^n IDF(q_i) \cdot \frac{f(q_i, D) \cdot (k + 1)}{f(q_i, D) + k \cdot (1 - b + b \cdot \frac{|D|}{avgdl})} \quad (4-8)$$

where query Q contains a set of keywords $\{q_1, q_2, \dots, q_n\}$, $score(D, Q)$ is the relevance score between query Q and document D , $IDF(q_i)$ is inverse document frequency weight of query term q_i . $f(q_i, D)$ denotes the frequency of query term q_i in document D . k and b are scalar parameters, usually chosen, in absence of an advanced optimization, as $k \in [1.2, 2.0]$ and $b = 0.75$. In this experiment, we used $k = 1.2, b = 0.75$. $avgdl$ denotes the average document length.

Each query was passed to the Terrier retrieval system to obtain the initial retrieval results for this query. The initial retrieval results for the query were later combined with the prediction result obtained from the recommender component in the integrated model.

4.3.2 Recommender Component

For the recommender component, the first step was to find the appropriate topic category for the current query, then to generate the recommendation for this query based on the information included in the selected topic category.

4.3.2.1 Problem of Categorizing without Centroid Representation

In this section, we use the current query and the statement for each topic category to categorize the query to a topic category.

- We represent both the current query and each topic as vectors. The query is represented as $\vec{q} = \{t_1, t_2, \dots, t_m\}$ and the topic statement as $\vec{T} = \{T_1, T_2, \dots, T_m\}$.
- The cosine similarity is then used to measure how similar the query is to each topic.

$$similarity(\vec{q}, \vec{T}) = \frac{\sum_{i=1}^m t_i \times T_i}{\sqrt{\sum_{i=1}^m (t_i)^2} \times \sqrt{\sum_{i=1}^m (T_i)^2}}$$

The results show that for 123 training queries, 79 of them find the corresponding correct topic categories. However, this result arises because the statement for both the query and the topic are quite short. The user's query may not overlap with many or possibly any terms in the topic categories, in this case, the retrieval result returned to the users may be extremely bad. In order to address this problem, we used a method to generate the centroid document (Radev et al., 2008) representing both the current query and each topic category. The centroid document is a summary of multiple documents. We chose this method to expand both the query and topic statement to address the mismatch problem caused by the short length of the query and topic statements. Finally, we compared the similarity between the representation of the current query and each topic category.

Different from the categorization procedure for the INEX and TREC-8 experiments in the previous chapter, which only generated a single representation for each topic category and compared the similarity between the query and the representation of each topic category, for experiments in this chapter, we generated a centroid representation for both the current query and the topic categories, and compare the centroid representations to select the best topic category. The following subsections introduce the process of generating the centroid representations for both the query and each topic category.

4.3.2.2 Generation of Centroid Representation

Generation of Centroid Document for Query

The centroid document for the current query q , was generated using the initial retrieval result for the query obtained using the IR component. The initial retrieval results were used to expand the query by taking top N documents in the ranked list to generate the centroid representation for this query. The top N documents were represented as a summed vector. For each document d in the

initial retrieval list, stopwords were first removed with subsequent application of Porter stemming. The resulting document vector was then weighted using (TF-IDF) to produce a weighted vector $d_{tf-idf} = (tf-idf_1, tf-idf_2, \dots)$, where $tf-idf_i$ is the term-frequency inverse-document-frequency of the i^{th} term. For the set N of documents and their corresponding vector representations, we defined the centroid vector C_q for the query using Equation (4-9).

$$C_q = \frac{1}{|N|} \cdot \sum_{i=1}^N d_i \quad (4-9)$$

where C_q is the centroid vector for the query q and d_i is the document in the top N subset of initial retrieval list. C_q is computed as the sum over the top N ranked documents divided by $|N|$.

Generation of Centroid Document for Topic Category

To generate the centroid document for each topic category, we used a procedure similar to that used in the TREC-8 experiment described in previous chapter:

- Compute the document frequency (df) for each document in each topic category, i.e. (df) means how many users viewed this document in this topic category. Then rank all documents in each topic category in descending order of their frequency.
- Choose the top M highest document frequency items in each category to generate the centroid representation for the corresponding topic category.
- Use Equation (4-10) to generate the centroid document for each topic category based on the selected M highest frequency documents in the corresponding topic category.

$$C_T = \frac{1}{|M|} \cdot \sum_{i=1}^M d_i \quad (4-10)$$

where C_T is the centroid document for a topic category and d_i is an item in the top M highest document frequency subset.

4.3.2.3 Categorizing with Centroid Representation

The purpose of this step is to attempt to identify the correct topic category for an input query. The process of selecting the best matching topic category for the current queries based on the centroid representation was as follows:

- Representations for both current query and each topic category were represented by a list of terms weighted by TF-IDF, the vectors were $C_q = (t_{1_{tf-idf}}, t_{2_{tf-idf}}, \dots, t_{n_{tf-idf}})$ and $C_T = (t_{1_{tf-idf}}, t_{2_{tf-idf}}, \dots, t_{m_{tf-idf}})$, where C_q is the centroid document for the current query and C_T is the centroid document for each topic category.
- Match the query representation (query centroid document) to each topic category by using cosine function, shown in Equation (4-11), to compute the distance between them. The closest topic category was selected as the most likely topic category for the current query q .

$$Similarity(C_q, C_T) = similarity_{cosine}(C_q, C_T) = \frac{C_q \cdot C_T}{\|C_q\| \|C_T\|} \quad (4-11)$$

As introduced above, the representation for both the current query and each topic category was the key factor in deciding whether the query was correctly matched to the most appropriate topic category. As mentioned, we used 27 randomly selected queries as the test query and the other 123 queries as training queries. For the 123 training queries, we examined using differing numbers of documents to generate the centroid document for them and each topic category. Results of this investigation are shown in Table 4.4.

<div style="display: flex; align-items: center; justify-content: center;"> <div style="writing-mode: vertical-rl; transform: rotate(180deg);">N</div> <div style="writing-mode: vertical-rl; transform: rotate(180deg);">M</div> </div>		Top M high frequency documents in selected topic category to generate centroid document for the topic category					
		M=2	M=3	M=4	M=5	M=7	M=10
Top N documents from initial rank list to generate centroid document for query	N=3	81	83	94	96	94	80
	N=5	87	89	107	112	104	93
	N=7	90	90	107	112	106	91
	N=10	95	95	109	113	104	93
	N=15	87	86	102	105	100	82
	N=20	81	82	99	98	94	79

Number of training queries finding the correct topic categories

Table 4.4 Accuracy of selecting the appropriate topic category by using different numbers of documents to generate centroid documents for both current query and topic categories for training set.

From Table 4.4, we can see that taking the top 10 documents from the initial ranking ($N = 10$) to generate the representation for the current query, while taking the 5 highest frequency documents ($M = 5$) to generate the representation for the topic category, obtains the best categorization results.

The reason that the correct topic category is not selected on some occasions is that we only used the 5 highest frequency documents to generate the centroid document for each topic category. Unfortunately, sometimes too many noisy documents are present in each topic category, such as non-relevant documents at high rank. Most users viewed these documents but

with low dwell time. In this case, only using the highest frequency documents to build the centroid documents for topic categories may lead to topic drift. We plan to examine methods which address this problem in future work.

Finally, 112 out of the 123 of the development training queries were matched to the correct topic category by using the centroid representations to do the categorizing. Compared to computing the similarity between the statement of query and each topic directly, which resulted in 79 out of 123 queries finding the correct topic category, the category assigned increased by approximately 28%.

4.3.2.4 Recommendation

This section describes the procedure for using the alternative recommender algorithms to compute the prediction output by exploiting the extended FIRE 2011 test collection described in Section 4.2.

Collection of Current User Search Behaviour

A number of the RS algorithms introduced in Section 2.2 require rating information for the current query to compute predictions, including item-based CF, cluster-based CF, rating-based CF and tendency-based CF. Our experiment suffers from the user-side cold start problem, which means users always only enter a query to the search engine and wait for the retrieved results. At this stage we have no method to acquire knowledge of these users to improve the effectiveness of the search results. To address this problem, similar to pseudo relevance feedback (PRF) in IR, we apply the query to the Terrier system utilizing the BM25 retrieval model to obtain a ranked list. The top N documents and their matching score obtained from the retrieval results are employed as ratings for the current query. From Table 4.4, we observe that using the top 10 documents to

generate the centroid documents obtains the best result. Following this result, we assumed that the top 10 documents were relevant to current query, and set N equal to 10.

Content-based filtering (CBF)

For the CBF method, the similarity between documents and the current query needs to be computed. Similar to the procedure for categorizing the current query to the appropriate topic category, the length of the query is too short to compute the current query and documents similarity reliably. To address this problem, we again used the centroid representation for the current query to do the similarity computation. The procedure for application of the CBF method to output recommendation for each test query was as follows:

- Generate the centroid document using the same procedures for current query, as introduced in Section 4.4.2.2.
- A set S is built which includes the retrieved documents for the current query and all previous users (users' clicked documents in all topics) rated documents.

$$S = S_{IR} \cup S_{TC}$$

where S_{IR} denotes the set of the ranked documents in the retrieval list and S_{TC} refers to the set of documents which have been rated by previous users. Represent each document in this set S as a vector i_{tf-idf} .

- For each query, compute the similarity between the representation of the query (C_q) and each document, and rank all documents in set S in descending order based on their distance from C_q .

Item-Based Collaborative Filtering (IBCF)

The procedure for using IBCF for each query was as follows:

- 1 Build an item-item similarity matrix: As mentioned in Section 2.2.2.2, since in our test collection we record all the previous users' search activities, including the items they viewed and associated ratings of each viewed item (using the dwell time as the rating information), we employed the adjusted cosine similarity (Equation 4-12) (which uses the ratings to compute the similarity between items) to compute the similarity between each pair of items, as shown in Figure 4.2.

$$S(i, j) = \frac{\sum_{u \in U} (v_{ui} - \bar{v}_u)(v_{uj} - \bar{v}_u)}{\sqrt{\sum_{u \in U} (v_{ui} - \bar{v}_u)^2 \sum_{u \in U} (v_{uj} - \bar{v}_u)^2}} \quad (4-12)$$

where $S(i, j)$ denotes the similarity between item i and j , v_{ui} is the rating user u gave to item i , and \bar{v}_u is the average rating of user u .

item i_1 to item i_{m-1}

	i_1	i_2	$i_3 \cdots i_{m-1}$	i_m		
item i_1 to item i_{m-1}	i_1	S_{11}	S_{12}	$S_{13} \cdots S_{1(m-1)}$	S_{1m}	<div style="display: flex; align-items: center;"> <div style="font-size: 3em; margin-right: 10px;">}</div> <div> S_{ij} denotes the similarity between item i and item j </div> </div>
	i_2	S_{21}	S_{22}	$S_{23} \cdots S_{2(m-1)}$	S_{2m}	
	i_3	S_{31}	S_{32}	$S_{33} \cdots S_{3(m-1)}$	S_{3m}	
	\cdots	\cdots	\cdots	\cdots	\cdots	
	i_{m-1}	$S_{(m-1)1}$	$S_{(m-1)2}$	$S_{(m-1)(m-1)}$	$S_{(m-1)m}$	
	i_m	S_{m1}	S_{m2}	$S_{m3} \cdots S_{m(m-1)}$	S_{mm}	

}

S_{ij} is the similarity between item i and item j

Figure 4.2 Example of item-item similarity matrix

- 2 Compute prediction: For each of the 27 test queries, we computed the retrieval list. The associated matching scores of the documents in this list were used as the click-through data for this query. The prediction of an item i for the current user u was then computed as the sum of the ratings given by the user for items similar to item i .

Each rating was weighted by the corresponding similarity $s_{i,j}$ between items i and j , where j is the document in the current user click-through document list.

$$P_{u,i} = \frac{\sum_j (S(i,j) v_{uj})}{\sum_j |S(i,j)|} \quad (4-13)$$

Cluster-Based Collaborative Filtering (CBCF)

Using the cluster-based CF method to generate the prediction list for a query proceeded as follows:

- 1 Fill topic categories: As introduced in section 3.2, each topic category contained m similar interest user search behaviours. Each user's search behaviour included a list of documents and the ratings that this user gave to each document. In this experiment, for each topic category, we filled each user's search behaviour by the category mean, that is, the items that a user had not rated were approximated as the mean of the ratings of these items obtained from the other users in the same topic category. For example, suppose a topic category contains 3 users' (user A , B and C) search behaviour, while user A rate items A, B and D, user B rate items A, B, C and E, and user C rate items B, C and D, see Figure 4.3(a). In order to fill user A 's search behaviour, the rating that user A may have given to item C was computed as the mean of the ratings of item C obtained from user B and C : $R_{AC} = \frac{1.4+2.0}{2} = 1.7$, and the rating user A gave to item E was $R_{AE} = \frac{2.1}{1} = 2.1$. The same computation method was applied to fill user B and C 's search behaviour profile, see Figure 4.3(b).

Items in the topic category	A	B	C	D	E
user A	1.4	3.5		2.6	
user B	2.0	2.8	1.6		2.1
user C		2.0	1.8	4.2	
(a)					
user A	1.4	3.5	1.7	2.6	2.1
user B	2.0	2.8	1.6	3.4	2.1
user C	1.7	2.0	1.8	4.2	2.1
(b)					

Figure 4.3 Example of filling each user's profile in the topic category using the cluster mean.

- 2 Topic category selection: For each of the 27 test queries, in the same way as described in section 4.3.2.3, the cosine function was used to select the appropriate topic category by computing the similarity between centroid representations of the query and each topic category.
- 3 Prediction computation: the users' search behaviour information in the selected topic category was used to compute the prediction of the current query by computing the mean value of each item in this topic category.

Tendency-Based Collaborative Filtering (TBCF)

- 1 For each test query, we first built the current user u 's search behaviour made for the query, and then again computed the mean rating

$$\overline{v_u} = \frac{\sum_{i \in \text{user search item list}} v_{ui}}{\text{number of items in user search list}}$$

for the current query, where v_{ui} is the rating which current user u gave to item i .

- 2 The same categorizing method was used here. Again the centroid documents were generated for both current query and each topic category. Then the cosine function was used to categorize the current query into a topic category by computing the similarity between their centroid representations.
- $\bar{v}_i = \frac{\sum_{u \in \text{users rate item } i \text{ in selected topic category}} v_{ui}}{\text{number of users rate item } i}$ the mean value for each item in the selected category was then computed.

- 3 The user tendency $\tau_u = \frac{\sum_{i \in I_u} (v_{ui} - \bar{v}_i)}{|I_u|}$ was computed. This is the average difference between the current user's ratings and the item mean.

- 4 The item tendency $\tau_i = \frac{\sum_{u \in U_i} (v_{ui} - \bar{v}_u)}{|U_i|}$ was computed based on the current user's search behaviour and the previous users' search behaviour in the selected topic category.

- 5 In order to compute the prediction of a rating that a user will give to an item, we estimated the characteristics of both the user tendency and the item tendency. This indicated whether the user and the item had positive tendency or negative tendency. The characteristic of user tendency was defined by comparing it with the target item mean value, and characteristic of item tendency was compared with the user mean rating. The four conditions of user and item tendency are listed in Table 4.5.

	User Tendency	Item Tendency
Condition 1	Positive ($\tau_u > \bar{v}_i$)	Positive ($\tau_i > \bar{v}_u$)
Condition 2	Negative ($\tau_u < \bar{v}_i$)	Negative ($\tau_i < \bar{v}_u$)
Condition 3	Negative ($\tau_u < \bar{v}_i$)	Positive ($\tau_i > \bar{v}_u$)
Condition 4	Positive ($\tau_u > \bar{v}_i$)	Negative ($\tau_i < \bar{v}_u$)

Table 4.5 Condition of user tendency and item tendency

- 6 Finally, the prediction that the current user would give to the target item was generated according to the four types of relations between user tendency and item tendency. This was computed using the following method:

- If both the user and the item had a positive tendency, the prediction was computed as

$$P_{ui} = \max(\bar{v}_u + \tau_{.i}, \bar{v}_{.i} + \tau_u).$$

- If both the user and item had a negative tendency, the prediction was computed as

$$P_{ui} = \min(\bar{v}_u + \tau_{.i}, \bar{v}_{.i} + \tau_u).$$

- If the user tendency was negative and the item tendency was positive, the prediction was computed as

$$P_{ui} = \min(\max(\bar{v}_u, (\bar{v}_{.i} + \tau_u) \cdot \beta + (\bar{v}_u - \tau_{.i})(1 - \beta)), \bar{v}_{.i}).$$

- If the user tendency was positive and item tendency was negative, the prediction was computed as

$$P_{ui} = \bar{v}_{.i} \cdot \beta + \bar{v}_u(1 - \beta).$$

β was set to 0.6 in the last two cases. This value was set according to the experiments reported by Cacheda et al. (2011).

Rating-Based Collaborative Filtering (RBCF)

- 1 For each test query, the same method using the retrieval documents and their associated matching score was again used to build the search behaviour model for the current user for the query.
- 2 Similar to other recommender algorithms, cosine similarity was again used to categorize the current query into a topic category by comparing the centroid documents of both the query and each topic category.
- 3 For each item j in the selected topic category, if this item did not occur in the current user u 's search behaviour list, this item j 's prediction was computed based on the

current user's search behaviour information and the previous users' search data in the chosen topic category, using the rating-based CF Weighted Slope-One method. This means that the prediction score $P^{ws1}(u)_j$ for the current user u will give to item j was computed using: $P^{ws1}(u)_j = \frac{\sum_{i \in S(u)} (dev_{j,i} + u_i) card(S_{j,i})}{\sum_{i \in S(u)} card(S_{j,i})}$, where i belongs to the set of items which the current user u has rated, $dev_{j,i}$ is the deviation between item i and j , and $card(S_{j,i})$ is the number of users who rate both item i and j in the selected topic category.

4.3.2.5 *Combination*

Merging multiple ranked lists is a common method to improve the effectiveness of an IR system. For example, Sheldon et al. (2011) present a LamdaMerge method to combine multiple retrieval results obtained from different query reformulations. In this experiment, with the objective of improving the retrieval effectiveness, we use the RS prediction results to re-rank the information retrieval results by combining two ranked lists. Alternative fusion methods are investigated for the RS methods described in Section 4.3.

4.4 Results

Retrieval effectiveness was evaluated using Mean Average Precision (MAP) and precision at cut-off rank.

Two baselines were used in order to assess the effectiveness:

- Standard initial retrieval without any query expansion method. (BL)

- Standard initial retrieval with PRF query expansion by adding 9 terms from top 10 documents. These numbers were chosen using query expansion experiments which we describe in Chapter 6. (BL+QE)

Experimental results are shown in Tables 4.6 and 4.7. for the following runs: BL, BL+QE, Content-based filtering (CBF), Cluster-based CF (CBCF), Tendency-based CF (TBCF), Item-based CF (IBCF), Rating-based CF (RBCF).

	CombSUM	CombMAX	CombMIN	CombANZ	CombNBZ	CombNorm
BL	0.1225	0.1225	0.1225	0.1225	0.1225	0.1225
BL+QE	0.1498 (+22.3%)	0.1498 (+22.3%)	0.1498 (+22.3%)	0.1498 (+22.3%)	0.1498 (+22.3%)	0.1498 (+22.3%)
BL+CBF	0.1558 (+27.2%)	0.1550 (+26.5%)	0.1540 (+25.7%)	0.1620 (+32.2%)	0.1615 (+31.8%)	0.1594 (+30.1%)
BL+CBCF	0.1990 (+62.5%)	0.1882 (+53.6%)	0.1455 (+18.8%)	0.1862 (+52.0%)	0.1865 (+52.2%)	0.1926 (+57.3%)
BL+IBCF	0.1705 (+39.2%)	0.1535 (+25.3%)	0.1550 (+26.5%)	0.1890 (+54.3%)	0.1789 (+46.0%)	0.1852 (+51.2%)
BL+TBCF	0.1768 (+44.3%)	0.1550 (+26.5%)	0.1560 (+27.3%)	0.1950 (+59.2%)	0.1923 (+56.9%)	0.1521 (+23.4%)
BL+RBCF	0.1995 (+62.9%)	0.2000 (+63.3%)	0.1470 (+20.0%)	0.2006 (+63.8%)	0.2000 (+63.3%)	0.1998 (+63.1%)

Table 4.6 MAP of 8 different runs using 6 fusion methods.

Table 4.6 shows the MAP values for the two baseline results and for combinations of the standard IR output with the 5 existing recommender techniques respectively. The results show that for BL+QE and the BL+CBF method combination with the IR output achieves similar performance, which means that the content-based CF method does not work effectively in this integrated model on the FIRE data collection. The MAP when combining cluster-based and item-based methods with the IR results obtained average improvement around 49.4% and 40.4% respectively compare to the BL. Combining cluster-based and rating-based CF methods with IR achieved impressive results, where the rating-based method achieves the best output among all these methods.

The following table shows combination results for three schemes using the 5 different fusion operators. The three schemes are: the IR output (BL), the content-based filtering output (CBF) and one out of four CF methods respectively.

	CombSUM	CombMAX	CombMIN	CombANZ	CombNBZ	CombNorm
BL	0.1275	0.1275	0.1275	0.1275	0.1275	0.1275
BL+QE	0.1498 (+22.3%)	0.1498 (+22.3%)	0.1498 (+22.3%)	0.1498 (+22.3%)	0.1498 (+22.3%)	0.1498 (+22.3%)
BL+CBCF+CBF	0.2010 (+57.6%)	0.1993 (+56.3%)	0.1648 (+32.1%)	0.2017 (+58.2%)	0.1967 (+54.3%)	0.2013 (+57.9%)
BL+IBCF+CBF	0.1935 (+51.8%)	0.1763 (+38.3%)	0.1725 (+37.4%)	0.1990 (+56.1%)	0.2000 (+56.9%)	0.2008 (+57.5%)
BL+TBCF+CBF	0.1987 (+55.8%)	0.1765 (+38.4%)	0.1756 (+37.7%)	0.2057 (+61.3%)	0.2084 (+63.5%)	0.2067 (+62.1%)
BL+RBCF+CBF	0.2053 (+61.0%)	0.2157 (+69.2%)	0.1695 (+32.9%)	0.2150 (+68.6%)	0.2043 (+60.2%)	0.2045 (+60.4%)

Table 4.7 MAP of 7 runs, Baseline with content-based CF and one of other 5 CF methods in 6 fusion methods.

As noted above, in Table 4.6 results for BL+CBF do not improve the MAP of the integrated model compared to BL+QE. However, in Table 4.7, when we treat CBF results as a component of the integrated model and combine it with the output of both IR and one of the other CF methods, the results obtained improve significantly compared to the results in Table 4.6. This shows that combining multiple recommender outputs can improve the effectiveness of the integrated model. Results in Table 4.7 also show that BL+RBCF+CBF achieves the best results among other methods.

Tables 4.6 and 4.7 indicate that the CombANZ fusion method performs best among the six methods examined. The results presented in these two tables answer the following research questions.

- Can recommender techniques help IR systems to obtain better retrieval results?

According to the results we obtained in the initial investigation in the previous chapter and the more realistic investigation in this chapter, we can answer this question in the affirmative that exploiting recommender techniques to generate predictions and combining these results with the standard IR output can help a standard IR system to obtain better retrieval results in our proposed integrated model.

- Can user logs be utilized to benefit the integrated IR model?

User logs can be exploited by different recommender algorithms to generate a prediction ranking. Combining this output with IR results can improve the IR retrieval results. The results are very good and show this is a good way to exploit user logs in the integrated model.

- Which recommender algorithms are most suitable in this application?

From the results shown in Table 4.6 and Table 4.7, we can answer this question in the affirmative using the hybrid approach which combines CBF with the rating-based (Weighted Slope One) CF method with the standard IR results (BL+RBCF+CBF) benefits the integrated IR model most. The existing rating-based method outperforms other methods in our model because of the characteristics of our test collection. We recorded users' search activity and extracted the dwell time as the ratings, which are a good indicator of item relevance. Based on the ratings information we extracted, rating-based CF algorithms, which exploit users rating information to compute prediction, are the most suitable method to benefit our model.

4.5 Analysis of the Suitability of Collaborative Filtering Methods for the Integrated Model

In the previous section, we examined the use of several existing recommender algorithms in the integrated model. These included a content-based filtering algorithm, an item-based filtering algorithm, a cluster-based smoothing method, a tendency-based filtering algorithm and rating-based filtering algorithms. We found that, although all these RSs share fundamental features with IR, similar to the initial experiment with Slope-One algorithm in the Chapter 3, combining IR with RAs can improve the rank of the relevant documents in the ranked IR output. However, their goals are not exactly same. Evaluation of most IR system focuses on an objective relevance, i.e. the relevance of a document to a user's information need based on their entered query. However, objective relevance makes no sense in RSs. RSs recommend items based on the likelihood that they will meet a specific user's taste or interest. Thus recommender algorithms recommend popular and novel items to users rather than items which have the greatest chance of being relevant. Based on this analysis, we hypothesize that using standard recommender algorithms to generate the prediction results and use this recommendation to re-rank the IR output may not be an optimal method for the integrated model, because RS and IR systems have different goals. In this section, we first explore this hypothesis, and then based on this analysis, propose a simple adaptation of an existing recommender algorithm based on the rating-based CF method and investigate its effectiveness in the integrated IR model. This adaptation is designed to make it more suitable for the integrated retrieval model in the way it involves characteristics of IR into a standard recommender algorithm. This section first analyses the behaviour of existing RS algorithms, then describes our adapted algorithm and concludes with an experimental evaluation of its effectiveness.

4.5.1 Analysis of Rating-Based CF in the Integrated Model

From the results shown in Table 4.5 and Table 4.6, we can observe that besides the CBF method, for CF schemes, utilizing the Rating-Based Slope-One algorithm in the integrated model achieves the best results. However, on inspection it can be seen that a situation can arise when using the Weighted Slope-One algorithm in which a document rated only by a small number of users, but where all of these users give it a high rating, may obtain a higher prediction value than a document which is well rated by most users, but with a relatively lower value. For example, when using the Weighted Slope-One algorithm, if a user rates a document A as 0.2, and 20 users rate both document A and document B with resulting average deviation between the documents of 0.6, while 2000 users rate both document A and document C with average deviation of 0.55, then document B is preferred (0.8 vs. 0.75). This result is preferred by recommender systems because they aim to recommend the highest rated documents to users. However, in an IR system, if a document has been rated by the majority of users for a particular query, and the average rating for this document is higher than the average value, this document must have the potential to be relevant to this query. Based on this theory, we would expect that document C has more potential to be relevant than document A . From this example, we can see that using the Weighted Slope-One algorithm to output the prediction ranking and combine it with IR results in the integrated model may decrease the IR ranking in some situation.

4.5.2 Popularity-Focused Rating-Based Collaborative Filtering Method (PFRBCF)

After observing the problem of using the Weighted Slope-One algorithm in the integrated model, we propose a way to address this problem by adding IR characteristics into the Weighted Slope-One algorithm to make it more suitable for the integrated model. As described in experiments conducted in Section 4.3, we cluster users into different topic categories based on their different

topic interests. Each topic category contains search behaviour information for similar taste users. For each query, we know corresponding topic category and relevance assessment. Based on the data we collected, which includes each query, its corresponding topic category, the user behaviour data in the topic category and the relevance assessment for this query, we can observe that in the corresponding topic category for each query, if any document in this topic category has been rated by most users, and also its obtained ratings are relative higher, it is more relevant to this query. From this observation, we can assume that document frequency in one topic category (how many time this document occurs in this topic category) indicates how many users in this topic category have rated this document, and that average document ratings are two factors affecting the document relevance. We propose a modified method by adding these two factors into the existing Weighted Slope-One algorithm to improve its suitability for the integrated model. This relationship between these two factors and relevance is shown in Table 4.8 in descending order, F_i is the document frequency and \bar{R}_i is the average rating that item i received.

F_i	\bar{R}_i	Indicator of Relevance
High	High	High Relevance
High	Low	Less Relevance
Low	High	Less Relevance
Low	Low	Non Relevance

Table 4.8 The Relationship between F_i , \bar{R}_i and relevance

Based on this assumption described above, we propose a novel Popularity Focused Weighted Slope-One (PWS1) algorithm which uses this relationship to extend the existing rating-based WS1 algorithm. This new algorithm is expected to be more effective for our integrated model because it includes two IR system factors. Our new prediction algorithm is shown in Equation

(4-14). This adjusts the recommender score that user u will give to item j by adding a parameter

$\frac{\log F_i + k}{1 - \frac{\bar{R}_j}{1 + \bar{R}_j}}$ to the standard Weighted Slope-One.

$$P^{PWS1}(u)_j = P^{WS1}(u)_j \cdot \frac{\log F_i + k}{1 - \frac{\bar{R}_j}{1 + \bar{R}_j}} \quad (4-14)$$

where F_i is the document frequency for document i , and \bar{R}_i is its average ratings, k is a constant value which is used for the condition that when this predicted document's frequency is only 1 in a topic category there will be a zero value for the final prediction for this document. We utilize $1 - (\bar{R}_j / 1 - \bar{R}_j)$ and $\log F_i + k$ to control the contribution of document frequency and average rating of this document for the final prediction score.

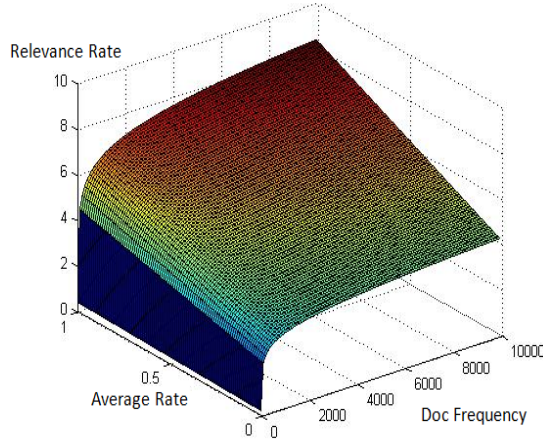


Figure 4.4 The trend of how document frequency and document average rate affect document relevance.

Figure 4.4. shows the relationship between document frequency and document average rating and the document relevance. These two factors work in combination with the original Weighted Slope-One algorithm when predicting the potential relevance for each document to smooth the effects of individual high ratings.

In the next section, the effectiveness of this popularity-focused rating-based CF algorithm is examined and compared to the other existing recommender algorithms in our integrated IR model.

4.5.3 Experimental Investigation

The results presented in Tables 4.6 and 4.7 reveal that, the hybrid recommender approach which incorporates CBF results with existing CF algorithms results outperforms the method which only combines the IR output with existing recommender algorithms results. Table 4.9 shows that MAP and precision at a cut-off position for the methods used in Table 4.7, and also compares these methods with our novel one which combines the IR output with CBF and the proposed PFRBCF by CombANZ fusion operator.

	MAP	P@5	P@10	P@20
BL	0.1275	0.1173	0.0967	0.0680
BL+QE	0.1498 (+22.3%)	0.1360 (+15.9%)	0.0890 (-7.96%)	0.0710 (+4.41%)
BL+CBCF+CBF	0.2017 (+58.2%)	0.1842 (+57.0%)	0.1400 (+44.8%)	0.0987 (+45.1%)
BL+IBCF+CBF	0.1990 (+56.1%)	0.1627 (+38.7%)	0.1340 (+38.6%)	0.0890 (+30.9%)
BL+TBCF+CBF	0.2057 (+61.3%)	0.1713 (+46.0%)	0.1373 (+41.9%)	0.0933 (+37.2%)
BL+RBCF+CBF	0.2150 (+68.6%)	0.1687 (+43.8%)	0.1343 (+38.9%)	0.0917 (+34.9%)
BL+PFRBCF+CBF	0.2231 (+75.0%)	0.1865 (+58.9%)	0.1520 (+57.2%)	0.1080 (+58.8%)

Table 4.9 MAP, P@5, P@10 and P@20 value for 7 runs on hybrid approach, merged by CombANZ fusion method.

From Table 4.9, it can be seen that our proposed PFRBCF algorithm obtains the best results, increasing MAP by 75% compared to the BL run, and by 50.74% compared to BL+QE when using the CombANZ method. MAP and precision at different cut off ranks for the hybrid approach show that our method achieves the greatest improvement for P@5, P@10 and P@20

compared to exploiting the other four existing recommender algorithms, CBCF IBCF TBCF and RBCF, to output prediction results to be combined with the IR output in the integrated model.

These results provide answers to our research question:

- Do existing recommender algorithms provide suitable solutions in this application?

Existing recommender algorithms can be used to benefit the integrated IR model, however, since the RSs and IR system have different goals, these existing algorithms are not ideally suited solutions in this application. According to the algorithm we proposed and the experimental results we obtained for this algorithm, we can conclude that adding features suitable for an IR application, such as the document frequency and the item's mean rating (which implicitly indicates the relevance of the item) into recommender algorithms, can make them more suitable for use in the integrated model and to achieve better results compared to the use of existing recommender algorithms.

4.6 Summary

In this chapter, we reviewed a range of existing recommender methods, including content-based filtering and collaborative filtering algorithms. We investigated their effectiveness in the integrated IR model. The results obtained demonstrate that they all can function as a recommender component to successfully improve IR search results for a query by utilizing the search history of previous users. However, IR systems and RSs have different objectives, where an IR system aims to find the most relevant items for a user, whereas RSs attempt to find items of interest to a user's tastes or interests. Based on this observation, we propose a simple modified popularity-focused rating-based recommender algorithm by incorporating the IR factors into an existing rating-based algorithm by incorporating relationship between the document frequency in

each topic category, document mean ratings and the document relevance grade. Results of the experimental evaluation of this new method in the integrated IR model demonstrate its effectiveness, which outperforms existing algorithms.

Besides investigating several recommender algorithms, in this chapter, we also reviewed six data fusion methods. They were examined in experiments combining the outputs of IR results and output of CF algorithm in the integrated model. The CombANZ fusion operator was found to achieve the best results. These fusion methods were also investigated in an experiment combining the outputs of IR, CBF algorithms and one of the CF algorithms. The CombANZ method again obtains the best results for these experiments.

Five existing recommender algorithms were examined for use directly to generate recommendation rank lists and in combination with the output of an IR component in the integrated model. The shortcomings of these methods were examined and a modified recommender algorithm introduced which was shown to improve the effectiveness of the integrated model. In next chapter, we aim to further examine if using recommender algorithms directly is the ideal method for the integrated model. To investigate this question, we attempt to exploit recommender techniques indirectly by using them in a cluster-based PageRank method.

Chapter 5

Exploiting Recommender Techniques in Cluster-Based Link Analysis

The basic idea of all experiments conducted in previous chapters is to employ recommender algorithms (RAs) to generate prediction results of items of potential interest to a user and to combine these results with the output of a standard IR system. However, as we have observed in the previous chapter, recommender systems (RSs) have fundamentally different goals to IR systems, and employing existing RAs may not be the best approach in the integrated model. Based on this observation, we make a hypothesis that instead of utilizing RAs in a straightforward way, we might take advantage of RAs by using them in an indirect method to aid the IR component in the integrated model. Since we already know that the PageRank algorithm has been widely proved to be an effective method in the IR system, in this chapter, we investigate the utilization of RAs in a cluster-based PageRank model to generate recommendations for the current query, and combine these recommendations with standard retrieval results in an alternative integrated model. We also propose a novel multiple cluster model by incorporating the current query into a model which looks into both the correlation between documents and the correlation between the cluster and documents. The chapter begins by considering how to employ this method in the integrated

model. This is followed by a description of the details of the methods to be used in this investigation.

5.1 Introduction

In the previous chapter, we examined the effectiveness of different recommender algorithms in an integrated IR model. In order to explore potentially more effective methods for exploiting recommender techniques in an integrated model, in this chapter we propose and investigate alternative methods of exploiting RAs in the integrated model.

PageRank is one of most popular algorithms for link analysis between web pages (Page et al., 1998) (Kurland and Lee, 2005). It is used by the Google web search engine to rank websites in their search engine results. PageRank provides a way of measuring the importance of web pages within the overall linked structure of the web. PageRank works by counting the number and quality of links to a web page to determine an estimate of its overall importance. The underlying assumption is that the more important a web page is on the web, the more likely it is to receive more links from other important web pages. PageRank values can be conceived as the steady state distribution of a Markov chain. More advanced web link analysis methods have been proposed to leverage the multi-layer relationships between web pages. The Conditional Markov Random Walk (MRW) model has been successfully applied in web page retrieval tasks based on a two-layer web graph (Liu and Ma, 2005). The hierarchical structure of the web graph is also exploited for link analysis in (Xue et al., 2005b), which is closely related to the link structure of the web. This relationship explains several important features of the web, including the locality and bidirectionality of hyperlinks, and compressibility of the web graph. Two existing link analysis models have been widely used in the PageRank method, the one-layer model and the two-layer model. In recent years, a number of researchers have focused on using link analysis

methods to re-rank search results in order to improve retrieval performance, e.g. Kurland and Lee (2005); Kurland and Lee (2006); Zhang et al. (2005). PageRank type methods have also been used successfully in other research, such as social network analysis (Zhou et al., 2007), multi-document summarization (Wan and Yang, 2008) and ad hoc search (Kurland and Lee, 2005). Based on these observations and analysis, we note that using either RSs or a PageRank algorithm can improve IR results. We propose a novel approach to improve standard IR result in this study by using RSs, PageRank and IR in a combined strategy by using RAs to compute the document distance in the PageRank algorithm. In the following sections, two RAs are examined in the two-layer model to compute the document importance ranking for the current query. Further a three-layer model is proposed based on the two-layer one which involves the current query to improve the effectiveness of the integrated model. The proposed methods are again evaluated using the expanded FIRE 2011 test collection introduced in previous chapter.

The structure of this chapter as follow: first we describe the two existing MRW models used in the PageRank algorithm. We then introduce the framework and methods we propose. We next describe the experiments conducted and their results, and finally draw conclusions for this investigation.

5.2 The PageRank Algorithm

In this section, two existing MRW models are described: a one-layer model and a two layer model. The application of these two models in the PageRank algorithm is then described.

5.2.1 Basic One-Layer Model

A MRW model is essentially a way of deciding the importance of a vertex within a graph based on global information recursively drawn from the entire graph. The basic idea is that of a “vote”

or “recommendation” between vertices (Wan and Yang, 2008). A link between two vertices is considered as a vote that one vertex gives to another. The score associated with a vertex is determined by the votes that are given for it.

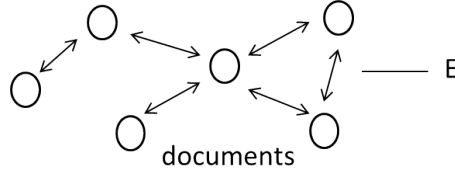


Figure 5.1 One-layer link graph (Wan and Yang, 2008)

Wan and Yang (2008) define notations in the following way. Given a document set S , let $G = (V, E)$ indicate a graph which reflects the relationships between documents in the whole document set, as shown in Figure 5.1. V is the set of vertices where each vertex v_i in V is a document in S . E is the set of edges, which is a subset of $V \times V$. Each edge e_{ij} in E is associated with an affinity weight $f(i \rightarrow j)$ between documents v_i and v_j ($i \neq j$). Each document v_i is represented as a set of terms $v_i(t_1, t_2, \dots, t_n)$. The affinity weight is computed using the standard cosine similarity measure (Baeza-Yates and Ribeiro-Neto, 1999) between two documents, shown in Equation (5-1).

$$f(i \rightarrow j) = \text{sim}_{\text{cosine}}(v_i, v_j) = \frac{\vec{v}_i \cdot \vec{v}_j}{|\vec{v}_i| \times |\vec{v}_j|} \quad (5-1)$$

where \vec{v}_i and \vec{v}_j are the term vectors of v_i and v_j . We think that two vertices are connected if the affinity weight between them is larger than 0. And define $f(i \rightarrow i) = 0$ is used to avoid the self-transition.

In Wan and Yang (2008), the transition probability matrix P , the transition probability from v_i to v_j $p(i \rightarrow j)$ is defined by normalizing the corresponding affinity weight as shown in Equation (5-2).

$$p(i \rightarrow j) = \begin{cases} \frac{f(i \rightarrow j)}{\sum_{k=1}^{|V|} f(i \rightarrow k)} & \text{if } \sum_{k=1}^{|V|} f(i \rightarrow k) \neq 0 \\ 0 & \text{otherwise} \end{cases} \quad (5-2)$$

Formally, $p(i \rightarrow j)$ is not equal to $p(j \rightarrow i)$. In (Wan and Yang, 2008), the authors use $(M_{i,j})_{|V| \times |V|}$ to describe G with each entry corresponding to the transition probability $M_{i,j} = p(i \rightarrow j)$. In order to make M into a stochastic matrix rows with all zero elements are replaced by a smoothing vector with all elements set to $1/|V|$. However, in our experiment, we are not concerned with the direction of documents, i.e. which document leads to which other document, which means that $p(i \rightarrow j)$ is equal to $p(j \rightarrow i)$. In our case, the saliency score $Score(v_i)$ for document v_i can be deduced from matrix M and formulated in a recursive form, as in the PageRank algorithm shown in Equation (5-3).

$$Score(v_i) = \lambda \cdot \sum_{j \neq i} Score(v_j) M_{j,i} + \frac{(1 - \lambda)}{|V|} \quad (5-3)$$

where λ is a damping factor usually set to 0.85, as in the PageRank algorithm (Page et al., 1998). For implementation, the initial scores of all documents are set to 1, and the iterative algorithm in Equation (5-3) is applied to compute the new scores of the documents. The convergence of the iteration algorithm is achieved when the difference between the scores computed for two successive iterations for any documents falls below a given threshold.

5.2.2 Two-Layer Model

A cluster-based conditional MRW model was proposed in (Wan and Yang, 2008). This conditional MRW model is based on a two-layer link graph including both documents and clusters information. This work assumed that a document set usually contains several non-related topics, that each top can be represented by a cluster of topic-related sentences, and that each

topic cluster is not equally important. The authors conducted experiments on the Document Understanding Conference (DUC) document summarization evaluation tasks dataset (DUC2001¹⁷ and DUC2002¹⁸ dataset). Three popular clustering algorithms were explored for detection of theme clusters within the document set: K-means Clustering, Agglomerative Clustering and Divisive Clustering. According to their results, the performance of each clustering algorithm varies based on the different λ value, shown in Equation (5-3). Overall, the Agglomerative Clustering algorithm obtained the best average performance among three clustering algorithms explored.

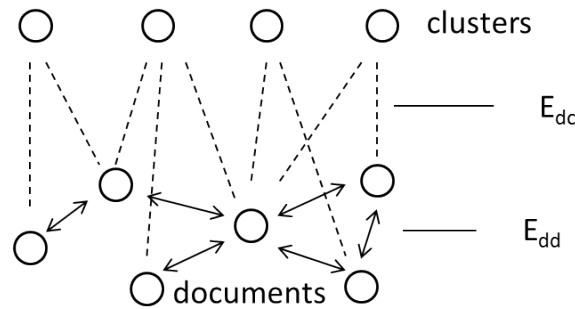


Figure 5.2 Two-layer link graph (Wan and Yang, 2008)

Wan and Yang (2008) also proposed the two-layer model is shown in Figure 5.2. The lower layer represents the traditional link graph between documents with the upper layer representing the topic clusters. The dashed lines between these two layers indicate the conditional influence between the documents and clusters. Formally, they represent the two-layer graph as $G^* = (V, V_c, E_{dd}, E_{dc})$ where V is the set of documents and V_c is the set of hidden nodes representing the detected theme clusters; $E_{dd} = \{e_{ij} | v_i, v_j \in V\}$ corresponds to all links between

¹⁷ <http://www-nlpir.nist.gov/projects/duc/guidelines/2001.html>

¹⁸ <http://www-nlpir.nist.gov/projects/duc/guidelines/2002.html>

documents and $E_{dc} = \{e_{ij} | v_i \in V, c_j \in V_c \text{ and } c_j = C(v_i)\}$ corresponds to the correlation between a document and its cluster. $C(v_i)$ indicates the theme cluster containing document v_i . They incorporated two factors, source cluster $C(v_i)$ and destination cluster $C(v_j)$, into the transition probability from v_i to v_j ; the new transition probability is defined as shown in Equation (5-4).

$$p(i \rightarrow j | C(v_i), C(v_j)) = \begin{cases} \frac{f(i \rightarrow j | C(v_i), C(v_j))}{\sum_{k=1}^{|V|} f(i \rightarrow k | C(v_i), C(v_k))} & \text{if } \sum_{k=1}^{|V|} f(i \rightarrow k | C(v_i), C(v_k)) \neq 0 \\ 0 & \text{otherwise} \end{cases} \quad (5-4)$$

where the $f(i \rightarrow j | C(v_i), C(v_j))$ is the affinity weight between two documents v_i and v_j , conditioned on the two clusters containing the two documents. $f(i \rightarrow j | C(v_i), C(v_j))$ is computed as shown in Equation (5-5).

$$\begin{aligned} f(i \rightarrow j | C(v_i), C(v_j)) &= \beta \cdot f(i \rightarrow j | C(v_i)) + (1 - \beta) \cdot f(i \rightarrow j | C(v_j)) \\ &= \beta \cdot f(i \rightarrow j) \cdot \pi(C(v_i)) \cdot \omega(v_i, C(v_i)) + (1 - \beta) \cdot f(i \rightarrow j) \\ &\quad \cdot \pi(C(v_j)) \cdot \omega(v_j, C(v_j)) \\ &= f(i \rightarrow j) \cdot \left(\beta \cdot \pi(C(v_i)) \cdot \omega(v_i, C(v_i)) + (1 - \beta) \right. \\ &\quad \left. \cdot \pi(C(v_j)) \cdot \omega(v_j, C(v_j)) \right) \end{aligned} \quad (5-5)$$

where $\beta \in [0,1]$ is the combination weight controlling the relative contributions from the source cluster and the destination cluster. In more precise detail, $\pi(C(v_i)) \in [0,1]$ denotes the importance of cluster $C(v_i)$ in the whole document set S . This aims to evaluate the importance of the cluster $C(v_i)$ in document set S , and is computed as the cosine similarity value between

the cluster and whole document set, shown in Equation (5-6), this equation is used to compute the similarity between the representation of the cluster $C(v_i)$ and the representation of document set S .

$$\pi(C(v_i)) = \text{sim}_{\text{cosine}}(C(v_i), S) \quad (5-6)$$

$\omega(v_i, C(v_i)) \in [0,1]$ denotes the strength of the correlation between document v_i and its cluster $C(v_i)$. This aims to evaluate the correlation between the document v_i and its cluster $C(v_i)$, and is computed as the cosine similarity value between the document and the cluster shown in Equation (5-7).

$$\omega(v_i, C(v_i)) = \text{sim}_{\text{cosine}}(v_i, C(v_i)) \quad (5-7)$$

The new row-normalized matrix \mathbf{M}^* is defined as shown in Equation (5-8).

$$M_{i,j}^* = p(i \rightarrow j | C(v_i), C(v_j)) \quad (5-8)$$

Similar to the one-layer model, the saliency score ($\text{Score}(v_i)$) for document v_i is computed based on the matrix \mathbf{M}^* by using the iterative form in Equation (5-3).

Based on these existing models, besides considering the correlation between clusters and documents in the PageRank algorithm, we propose a three-layer model which involves the correlation between query and cluster into the PageRank algorithm. The following sections introduce our proposed new algorithm.

5.3 Proposed Methods

In this section, we describe the framework of the integrated model exploiting the PageRank algorithm, and describe in detail the utilization of alternative recommender techniques in the two-layer model to improve the effectiveness of the integrated model. We then give details of

our proposed three-layer model by incorporating the query level, and investigate the effectiveness of exploiting RAs on this model.

5.3.1 Framework of Integrated Model Exploiting the PageRank Algorithm

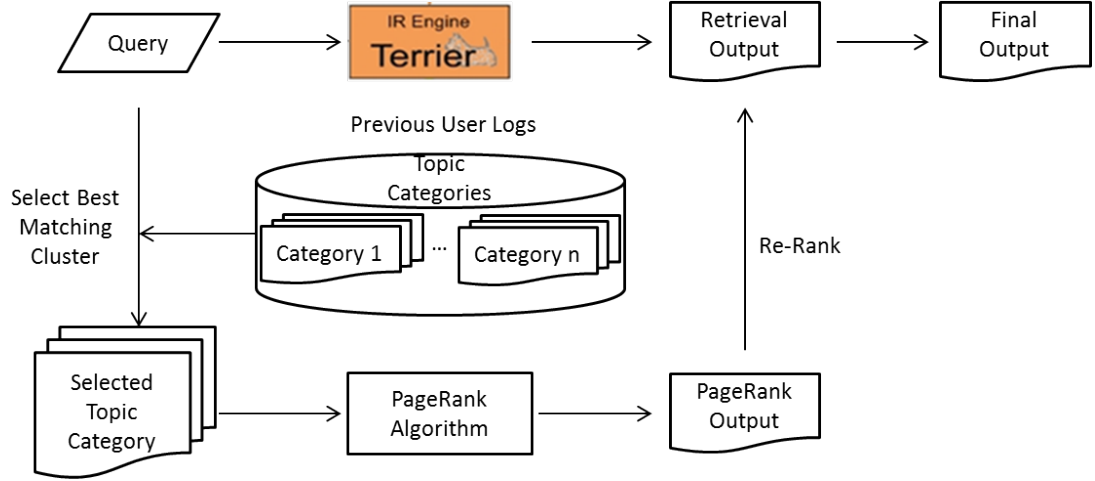


Figure 5.3 Workflow of integrated model using PageRank algorithm

Figure 5.3 shows the basic workflow of the integrated IR model using the PageRank algorithm to generate the document importance ranking. The combination of the PageRank output with the ranked IR results list is the last step of the process. The process is the same as that introduced in Chapter 3. The difference here is that instead of using a recommender technique, we use the PageRank algorithm to calculate the document importance ranking, and combine this with ranked IR output.

The “Previous User Logs” data set shown in Figure 5.3 contains a set of topic categories, similar to the previous system, where each topic category again includes user search logs of similar interests. Every user’s search log includes a list of viewed documents and the rating they gave to each document for each query. Since users may again be unwilling to provide explicit ratings for each viewed document, similar to the experiments in the previous chapter, we

calculate a document rating by extracting the dwell time that the user spent on each document. We assume that if any two documents are rated by the same user and their affinity weight is larger than 0 (affinity weight between two documents presents a measurement of how similar the documents are), that there is a connection between these two documents. This connection is used to build a link graph for each user in each topic category. Formally, as introduced in Section 5.2, PageRank detects the affinity weight between documents by computing a similarity between them, and exploits this affinity weight to predict each document's importance. In our method, we extract the dwell time for use as the rating value for each document. Since view time is a good indicator of the quality of documents, based on this theory, we use recommender techniques which exploit the rating value of each document to compute the similarity between documents, to extract affinity weight between two documents, and then use this extracted correlation to compute the importance of each document using the PageRank algorithm. Two recommender techniques which exploit users' ratings for prediction are investigated in this chapter: adjusted cosine similarity (Cacheda et al., 2011) and Weighted Slope-One (Lemire and Maclachlam, 2005).

To operate this model, we need to select the best matching topic category for each query. We observe that the category selection process introduces another affinity between the current query and every topic category. Based on this observation, besides using the two recommender techniques in the two-layer PageRank model, another two methods are proposed incorporating this query-cluster level affinity. Thus, a total of four methods are proposed. The details of these methods are introduced in the following sub-sections.

5.3.2 Using Adjusted Cosine Similarity in Two-Layer Model

Based on the cluster-based conditioned two-layer model introduced in Section 5.2.2, instead of using cosine similarity to compute the affinity weight between documents, in this section we propose a method exploiting adjusted cosine similarity (Cacheda et al., 2011), introduced in Section 2.2, to compute the affinity weight based on the document ratings. The adjusted cosine similarity is used to compute the similarity between two documents based on the ratings they received from different users for each document.

The affinity weight between documents is computed using Equation (5-9). Each topic category contains a number of weighted documents. The affinity weight between documents can simply be computed as the mean adjusted cosine similarity value in all topic categories which contain both document v_i and v_j based on all previous users search activities.

$$f(i \rightarrow j) = \frac{\sum_{C(v_i, v_j)} \frac{\sum_{u \in U} (v_{ui} - \bar{v}_u)(v_{uj} - \bar{v}_u)}{\sqrt{\sum_{u \in U} (v_{ui} - \bar{v}_u)^2 \sum_{u \in U} (v_{uj} - \bar{v}_u)^2}}}{card(C(v_i, v_j))} \quad (5-9)$$

where v_{ui} is the rating user u gives to document v_i , \bar{v}_u indicates the average rating of user u who rates both documents v_i and v_j in the selected topic category, U is a set of users in the corresponding topic category. $C(v_i, v_j)$ is the category containing both document v_i and v_j , and $card(C(v_i, v_j))$ denotes the number of topic categories which contain both document v_i and v_j .

For the current query q , similar to the previous chapters, we select its best matching topic category C_q , the transition probability from v_i to v_j is computed using Equation (5-10).

$$\begin{aligned}
& p(i \rightarrow j | C_q(v_i, v_j)) \\
&= \begin{cases} \frac{f(i \rightarrow j | C_q(v_i, v_j))}{\sum_{k=1}^{|V|} f(i \rightarrow k | C_q(v_i, v_k))} & \text{if } \sum_{k=1}^{|V|} f(i \rightarrow k | C_q(v_i, v_k)) \neq 0 \\ 0 & \text{otherwise} \end{cases} \quad (5-10)
\end{aligned}$$

where $f(i \rightarrow j | C_q(v_i, v_j))$ is computed using Equation (5-11).

$$\begin{aligned}
& f(i \rightarrow j | C_q(v_i, v_j)) \\
&= \beta \cdot f(i \rightarrow j) \cdot \omega(v_i, C_q) + (1 - \beta) \cdot f(i \rightarrow j) \cdot \omega(v_j, C_q) \quad (5-11) \\
&= f(i \rightarrow j) \cdot (\beta \cdot \omega(v_i, C_q) + (1 - \beta) \cdot \omega(v_j, C_q))
\end{aligned}$$

where $\beta \in [0,1]$ is the combination weight. Let $\omega(v_i, C_q) \in [0,1]$ indicate the strength of the correlation between document v_i and the topic category C_q , which is computed as $\omega(v_i, C_q) = \text{Sim}_{\text{cosine}}(v_i, C_q)$. The new row-normalized matrix \mathbf{M}^* is defined as shown in Equation (5-12).

$$M_{i,j}^* = p(i \rightarrow j | C_q(v_i, v_j)) \quad (5-12)$$

Finally, we compute the saliency score for each document and rank these documents in a descending order to be used as the output of the PageRank algorithm as shown in Figure 5.3. The saliency score for each document is computed based on the matrix \mathbf{M}^* by using the iterative form shown in Equation (5-13).

$$\text{Score}(v_i) = \lambda \cdot \sum_{j \neq i} \text{Score}(v_j) M_{j,i} + \frac{(1 - \lambda)}{|V|} \quad (5-13)$$

where λ denotes a damping factor usually set to 0.85, as in the PageRank algorithm (Page et al., 1998). For implementation, the initial scores of all documents are set to 1 and the iteration algorithm in Equation (5-13) is adopted to compute the new scores of the documents. The convergence of the iteration algorithm is achieved when the difference between the scores computed at two successive iterations for any documents falls below a given threshold. The threshold is set to 0.001 in this study. This threshold is set experimentally. We examined that

after the difference between two successive iterations for any documents falls below 0.001, if we still perform further iterations, there is little change in the document rank.

5.3.3 Using Adjusted Cosine Similarity in the Three-Layer Model

Following on from the previous section, we still exploit the adjusted cosine similarity algorithm to compute the affinity weight between documents based on the rating they received. The difference here is that instead of using the existing two-layer model, in this section, we propose a novel three-layer model in the PageRank algorithm. Since the correlation between the current query and the selected topic category is an important indicator of whether the appropriate topic category is selected which may affect the results a lot, we anticipate that this three-layer model can be more effective than using the standard one-layer or two-layer model in PageRank method in the integrated model.

Wan and Yang (2008) exploit the two-layer model in their experiments for multi-document summarization. Based on the good results they obtained and their conclusions, we observe that the relation between documents and the cluster they belong to can help to improve the effectiveness of computing the document importance. Also from our previous experiments, we can draw a conclusion that the query-to-cluster relationship is a good indicator of the relevance between documents and the current query. Based on these observations, we propose a three-layer model which incorporates query-level information and the query-to-cluster relationship with the existing two-layer model. The structure of three-layer model is shown in Figure 5.4. The lower layer is the traditional link graph between documents in the basic Markov Random Model (MRW) model. The link between documents contains two types of correlations $E_{dd} = \{e_{ij} | v_i, v_j \in V\}$ corresponds to the links between documents, and $N_{dd} = \{e_{ij} | card(u_{ij})\}$ is the number of users who rate both documents v_i and v_j . The middle layer represents the topic

categories (clusters). The dashed lines between lower and middle layers indicate the conditional influence between the documents and the topic categories. The upper layer is the query layer. The dashed lines between the query layer and the topic category layers indicate the strength of the correlations between queries and topic categories.

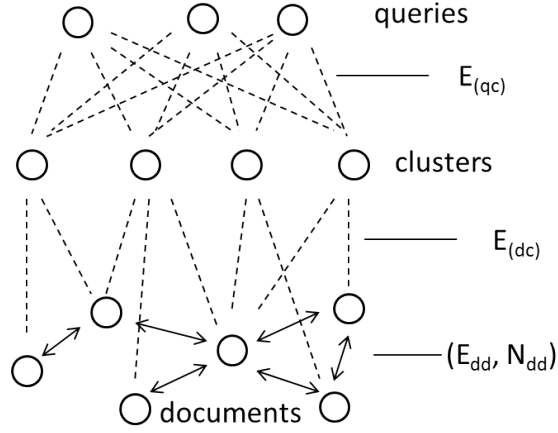


Figure 5.4 Three-layer link graph

Our hypothesis is that for any two documents always rated by the same user, there is strong correlation between them. Based on this hypothesis, we adopt $N(v_i, v_j)$ which denotes the number of users who rate both documents v_i and v_j in all topic categories, and include this relationship in computing the affinity weight for documents. Factor N_{dd} indicates the strength of the correlation between two documents, while $E_{(qc)}$ denotes the correlation between a query q_i and each topic category C . Then for the current query q , its best match topic category C_q is selected, the affinity weight between any two documents v_i and v_j is computed by using the adjusted cosine similarity as shown in Equation (5-14).

$$f(i \rightarrow j) = \frac{\sum_{C(v_i, v_j)} \frac{\sum_{u \in U} (v_{ui} - \bar{v}_u)(v_{uj} - \bar{v}_u)}{\sqrt{\sum_{u \in U} (v_{ui} - \bar{v}_u)^2 \sum_{u \in U} (v_{uj} - \bar{v}_u)^2}}}{card(C(v_i, v_j))} \times N(v_i, v_j) \quad (5-14)$$

So for the current query q , in the selected topic category C_q , the transition probability from document v_i to document v_j is computed using Equation (5-15).

$$p(i \rightarrow j|q, C_q(v_i, v_j)) = \begin{cases} \frac{f(i \rightarrow j|q, C_q(v_i, v_j))}{\sum_{k=1}^{|V|} f(i \rightarrow k|q, C_q(v_i, v_k))} & \text{if } \sum_{k=1}^{|V|} f(i \rightarrow k|q, C_q(v_i, v_k)) \neq 0 \\ 0 & \text{otherwise} \end{cases} \quad (5-15)$$

where $f(i \rightarrow j|q, C_q(v_i, v_j))$ is computed using Equation (5-16).

$$\begin{aligned} f(i \rightarrow j|q, C_q(v_i, v_j)) &= \beta \cdot f(i \rightarrow j) \cdot \pi(q, C_q) \cdot \omega(v_j, C_q) + (1 - \beta) \cdot f(i \rightarrow j) \cdot \pi(q, C_q) \\ &\quad \cdot \omega(v_j, C_q) \\ &= f(i \rightarrow j) \cdot \pi(q, C_q) \cdot (\beta \cdot \omega(v_i, C_q) + (1 - \beta) \cdot \omega(v_j, C_q)) \end{aligned} \quad (5-16)$$

where $\beta \in [0,1]$ is the combination weight, $\pi(q, C_q) \in [0,1]$ denotes the strength of the correlation between the current query q and the selected cluster C_q , and is computed as: $\pi(q, C_q) = \text{sim}_{\text{cosine}}(q, C_q)$ and $\omega(v_j, C_q) \in [0,1]$ represents the correlation between document v_j and the cluster C_q it belongs to, also computed by cosine similarity $\omega(v_i, C_q) = \text{sim}_{\text{cosine}}(v_i, C_q)$. The new row-normalized matrix M^* is defined as shown in Equation (5-17).

$$M_{i,j}^* = p(i \rightarrow j|q, C_q(v_i, v_j)) \quad (5-17)$$

$\text{Score}(v_i)$, the saliency score for each document, is computed based on the matrix M^* by using the following iterative form:

$$\text{Score}(v_i) = \lambda \cdot \sum_{j \neq i} \text{Score}(v_j) M_{j,i} + \frac{(1 - \lambda)}{|V|}$$

Similar to the previously described methods, λ is set to 0.85, the initial score of all documents is set to 1.0, and the threshold for the difference between two successive iterations for any documents is set to 0.001.

5.3.4 Using Document Deviation in Two-Layer Model

Similar to Section 5.3.2, instead of using the adjusted cosine similarity to compute the correlation between documents, in this section, we exploit another recommender algorithm (RA). The rating-based Weighted Slope-One Scheme is again selected to extract the correlations between documents based on the ratings assigned to them.

In this method, the affinity weight $f(i \rightarrow j)$ is computed as the deviation between documents v_i and v_j instead of similarity between two documents. This deviation is a measurement of the difference between documents, and is based on the mean average of the users' ratings for the documents. Based on the Weighted Slope-One scheme, both document deviation and the number of users who rate both document v_i and v_j are factors in representing the correlation between documents v_i and v_j . These factors are explored in this method to improve effectiveness of the MRW model. For our experiment, in order to compute the affinity weight between documents, we define all users' search activity information as the training set χ , and u_i and u_j as the ratings that a user u gives to documents v_i and v_j respectively (annotated as $u \in S_{ij}(\chi)$). We compute the affinity weight between document v_i and v_j (the same as computing the average deviation of document v_i with respect to v_j) using Equation (5-18).

$$f(i \rightarrow j) = dev_{v_i, v_j} = \sum_{u \in S_{ij}(\chi)} \frac{|u_j - u_i|}{card(S_{ij}(\chi))} \quad (5-18)$$

where $card(S_{i,j}(\chi))$ indicates the number of users who rate both document v_i and v_j in all topic categories.

The transition probability, row-normalized matrix and salience score for each document are computed in the same way as in Section 5.3.2. The new transition probability $p(i \rightarrow j|C_q(v_i, v_j))$ is computed using Equation (5-19).

$$p(i \rightarrow j|C_q(v_i, v_j)) = \begin{cases} \frac{f(i \rightarrow j|C_q(v_i, v_j))}{\sum_{k=1}^{|V|} f(i \rightarrow k|C_q(v_i, v_k))} & \text{if } \sum_{k=1}^{|V|} f(i \rightarrow k|C_q(v_i, v_k)) \neq 0 \\ 0 & \text{otherwise} \end{cases} \quad (5-19)$$

$f(i \rightarrow j|C_q(v_i, v_j))$ is computed using Equation (5-20).

$$\begin{aligned} f(i \rightarrow j|C_q(v_i, v_j)) &= \beta \cdot f(i \rightarrow j) \cdot \omega(v_i, C_q) + (1 - \beta) \cdot f(i \rightarrow j) \cdot \omega(v_j, C_q) \\ &= f(i \rightarrow j) \cdot (\beta \cdot \omega(v_i, C_q) + (1 - \beta) \cdot \omega(v_j, C_q)) \end{aligned} \quad (5-20)$$

The new row-normalized matrix \mathbf{M}^* and the saliency score for each document are defined and computed in the same way as in section 5.2.2:

$$M_{i,j}^* = p(i \rightarrow j|C_q(v_i, v_j)) \quad (5-21)$$

\mathbf{M}^* is also used in the PageRank algorithm (Equation (5-13)) to compute the saliency score for each document to obtain a descending ordered ranked list for the current query q . The PageRank results are fused with the retrieval results for query q using CombANZ operator in the final step to generate the final combined output for current query.

5.3.5 Using Document Deviation in Three-Layer Model

In this section, we investigate the effectiveness of utilizing the Weighted Slope-One algorithm in the three-layer model by the way of exploiting it for computing the weights between documents in the three-layer model. Besides this, we also consider three factors in this method: the weighted documents links, correlation between topic categories and documents, and the correlation between current query and the topic categories. All these factors are then given to the PageRank algorithm to generate the document importance rank. Finally, the PageRank output is combined with IR result to form the overall result.

Again, the affinity weight $f(i \rightarrow j)$ between document v_i and v_j is computed using document deviation is as shown in Equation (5-22).

$$f(i \rightarrow j) = dev_{v_i, v_j} = \sum_{u \in S_{ij}(\chi)} \frac{u_j - u_i}{card(S_{ij}(\chi))} \quad (5-22)$$

The new transition probability $p(i \rightarrow j|q, C_q(v_i, v_j))$ is computed using Equation (5-23).

$$\begin{aligned} M_{i,j}^* &= p(i \rightarrow j|q, C_q(v_i, v_j)) \\ &= \begin{cases} \frac{f(i \rightarrow j|q, C_q(v_i, v_j))}{\sum_{k=1}^{|V|} f(i \rightarrow k|q, C_q(v_i, v_k))} & \text{if } \sum_{k=1}^{|V|} f(i \rightarrow k|q, C_q(v_i, v_k)) \neq 0 \\ 0 & \text{otherwise} \end{cases} \end{aligned} \quad (5-23)$$

The new row-normalized matrix $M_{i,j}^* = p(i \rightarrow j|q, C_q(v_i, v_j))$ and the saliency score for document v_i is computed using Equation (5-24).

$$Score(v_i) = \lambda \cdot \sum_{j \neq i} Score(v_j) M_{j,i} + \frac{(1 - \lambda)}{|V|} \quad (5-24)$$

The iterative PageRank algorithm is used to compute the saliency score for each document, and finally output a document list ranked by importance based on these scores. Finally, the

PageRank output is combined with the retrieval results and the fusion results are returned for the current query.

5.4 Experimental Investigation

This section describes our experimental investigation examining exploration of different recommender techniques in the PageRank algorithm to generate document importance rank, and its combined action with the standard IR output. The extended FIRE 2011 test collection was again used for experiments conducted in this chapter. The following sub-sections describe four aspects of this experimental investigation: the procedure used to set up the IR and PageRank components for these experiments, the use of this test collection in the PageRank component, the fusion method and the results obtained.

5.4.1 Information Retrieval Component

The IR component used for this investigation is same as that used in Chapter 4, using the BM25 retrieval model in the Terrier system to generate a retrieval ranked list. This ranked output is combined with the PageRank component output in the final step.

5.4.2 PageRank Component

This section describes the experimental investigation into the PageRank component, including the process of exploiting the proposed methods in the PageRank algorithm to generate a document importance result list, and the method of using user logs to conduct these experiments.

The generation procedure of the PageRank component was:

- 1 Generate a centroid document for both the current query and each topic category.
- 2 Categorize the current query to the best matching topic category.

- 3 Build a link graph based on the users' search behaviours in the selected topic category.
- 4 Exploit a recommender algorithm to compute the correlation between documents according to the users' ratings and the link graph in the selected topic category.
- 5 Use the PageRank algorithm to generate a document importance ranked list.

The centroid document generation and categorization steps were the same as those described in previous chapters, Sections 4.3.2.2 and 4.3.2.3. Thus details of the procedure are not provided again in this chapter. The following sections report the procedures for steps 3 to 5 which exploited alternative recommender techniques in the various layers of the model in the PageRank algorithm to compute the document importance rank.

5.4.2.1 Build link graph

Information of the users' search behaviour was processed to build a link graph for each topic category. Our assumption was that in each topic category, if any two documents had been rated by the same user and the affinity weight between them is greater than 0, then there was a correlation between them and a link was built between them. Each topic group can be seen as a topically focused category. Every topic category can generate a user browsing graph like that shown in Figure 5.3. The different line types in Figure 5.4. represent the link graph for different users.

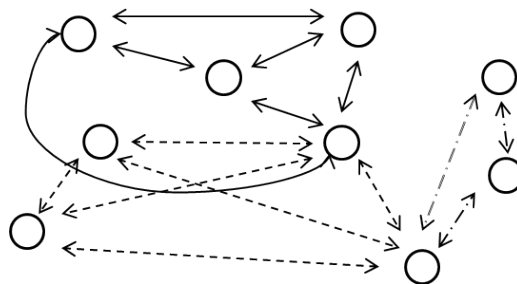


Figure 5.5 Sample of user browsing graph in one topic category

5.4.2.2 Exploiting Recommender Techniques in the PageRank Algorithm

After building the link graph of the selected topic category for the current query as represented in Section 5.3, two recommender techniques, adjusted cosine similarity and document deviation, were used on both the two-layer and three-layer models for the PageRank algorithm.

Adjusted cosine similarity in two-layer model (abbreviate: Acos_PR)

For the current query q , after selecting the best matching topic category (C_q) and building the link graph, the following steps were used to generate the document importance rank using the adjusted cosine similarity technique in the two-layer model in the PageRank component.

- 1 Based on the link graph, compute affinity weight ($f(i \rightarrow j)$) between every two documents v_i and v_j ($v_i, v_j \in C_q$) using the mean adjusted cosine similarity (Equation (5-9)).
- 2 Compute the correlation between the selected topic category and each document in it using cosine similarity as $\omega(v_i, C_q) = Sim_{cosine}(v_i, C_q)$ equation.
- 3 Compute the transition probability ($p(i \rightarrow j | C_q(v_i, v_j))$) between each document pair is based on the document affinity weight $f(i \rightarrow j)$ and the strength between the selected topic category and each document in this topic category $\omega(v_i, C_q)$ (Equation (5-10)).
- 4 Compute the saliency score for each document iteratively using the PageRank algorithm (Equation (5-13)).
- 5 Rank documents in C_q in descending order according to their saliency score, this rank is the outcome of PageRank component.

Adjusted cosine similarity in three-layer model (Abbreviate: CAcos_PR)

Using the adjusted cosine similarity method in the three-layer model is similar to its use in the two-layer model. The difference is that in the case of the three-layer model, we also consider the correlation between the current query and its selected topic category.

- 1 Based on the link graph, compute the affinity weight ($f(i \rightarrow j)$) between every each document pair v_i and v_j ($v_i, v_j \in C_q$) using the mean adjusted cosine similarity (Equation (5-14)).
- 2 Compute the correlation between the selected topic category and each document in it using the cosine similarity method, as $\omega(v_i, C_q) = Sim_{cosine}(v_i, C_q)$.
- 3 Calculate the strength of correlation between the current query q and the selected cluster C_q as $\pi(q, C_q) = sim_{cosine}(q, C_q)$.
- 4 Compute the transition probability ($p(i \rightarrow j|q, C_q(v_i, v_j))$) between each document pair based on the document affinity weight $f(i \rightarrow j)$, query and cluster correlation ($\pi(q, C_q)$) and the strength of the correlation between selected topic category and each document in this topic category $\omega(v_i, C_q)$ (Equation (5-15)).
- 5 The saliency score for each document is computed iteratively using the PageRank algorithm (Equation (5-13)).
- 6 Documents in C_q ranked in descending order according to their saliency score as the output of the PageRank component.

Document Deviation in Two-Layer Model (Abbreviate: Dev_PR)

Instead of using the adjusted cosine similarity as in the two-layer model, this method exploits the document deviation technique to compute the correlation between every document pair in the link graph in two-layer model. The outcome generation procedure is as follows:

- 1 Based on the link graph, compute the affinity weight ($f(i \rightarrow j)$) between each document pair v_i and v_j ($v_i, v_j \in C_q$) using document deviation computation (Equation (5-18)).
- 2 Compute the correlation between the selected topic category and each document in it using the cosine similarity method, as $\omega(v_i, C_q) = Sim_{cosine}(v_i, C_q)$.
- 3 Compute the transition probability ($p(i \rightarrow j | C_q(v_i, v_j))$) between each document pair based on the document affinity weight $f(i \rightarrow j)$ and the strength between selected topic category and each document in this topic category $\omega(v_i, C_q)$ (Equation (5-10)).
- 4 Compute the saliency score for each document iteratively using the PageRank algorithm (Equation (5-13)).
- 5 Documents in C_q are ranked in descending order according to their saliency score as the output of the PageRank component

Document Deviation in Three-Layer Model (Abbreviate: CDev_PR)

The method of exploiting document deviation in the three-layer model to compute the document correlation is similar to using document deviation in two-layer model. The difference is that here we also consider the correlation between query and the topic category selected for it.

- 1 Based on the link graph, compute the affinity weight ($f(i \rightarrow j)$) between each document pair v_i and v_j ($v_i, v_j \in C_q$) by means of the document deviation computation (Equation (5-18)).
- 2 Compute the correlation between the selected topic category and each document in this topic category using the cosine similarity method, by $\omega(v_i, C_q) = Sim_{cosine}(v_i, C_q)$.

- 3 Calculate the strength of correlation between the current query q and the selected cluster C_q by $\pi(q, C_q) = \text{sim}_{\text{cosine}}(q, C_q)$.
- 4 Compare the transition probability ($p(i \rightarrow j | q, C_q(v_i, v_j))$) between each document pair based on the document affinity weight ($f(i \rightarrow j)$), query and cluster correlation ($\pi(q, C_q)$) and the strength of correlation between selected topic category and each document in this topic category $\omega(v_i, C_q)$ (Equation (5-15)).
- 5 The saliency score for each document is computed iteratively using the PageRank algorithm (Equation (5-13)).
- 6 Documents in C_q are ranked in descending order according to their saliency score as the output of the PageRank component.

After using the adjusted cosine similarity and document deviation recommender techniques in both the two-layer and three-layer models to compute the affinity weight between documents used in PageRank algorithm to generate the document importance rank, the output of the PageRank component was combined with the IR component output. The following section introduces the combination methods used in this experiment.

5.4.2.3 Component Fusion in the Integrated Model

The investigation of alternative fusion operators in the last chapter showed that the combANZ fusion operator obtains the best results for combining IR results with prediction ranking. Thus in this experiment, we use the combANZ operator to combine the results. Let S_{IR} refer to the retrieval list and S_{PR} denote the novel PageRank output list. For the combANZ method, each document's new weight is calculated as shown in Equation (5-19).

$$d'_{\text{combANZ}} = \begin{cases} (d_{IR} + d_{PR})/2 & \text{if } (d \in S_{IR} \cap S_{PR}) \\ d_{IR} & \text{otherwise} \end{cases} \quad (5-19)$$

where d_{IR} refers to the relevance score of document d in S_{IR} and d_{PR} indicates the importance score of document d in S_{PR} .

5.4.3 Experimental Results

Retrieval effectiveness is evaluated in terms of Mean Average Precision (MAP) and precision at cut-off rank n . Four different baselines are compared with the cluster-based PageRank methods proposed in Section 5.2. The three baselines are:

- 1 Standard ranked information retrieval (IR) using Terrier search engine, BM25 retrieval model.

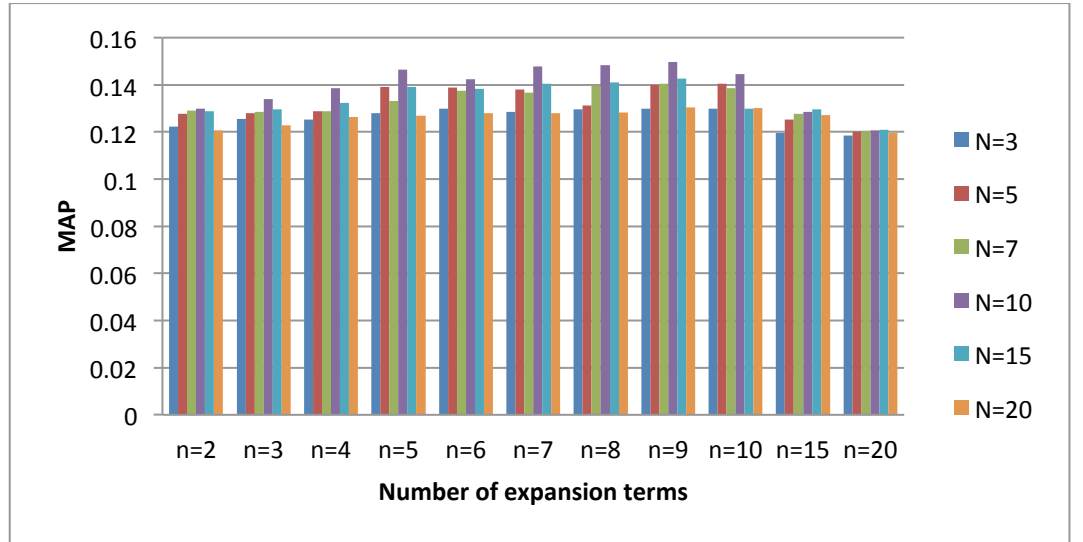


Figure 5.5 MAP of choosing vary number of documents and additional terms for PRF query expansion.

- 2 Query expansion using Terrier Bo1 pseudo-relevance feedback (PRF) on the initial retrieval results, using the top N documents in the initial retrieval list, adding n terms to the test query. Figure 5.6 shows the results of choosing different numbers of top ranked documents to carry out query expansion and adding varying numbers of terms to the current query on the training data. From the figure we can see that choosing the

top 10 documents and adding 9 additional terms to the current query obtains the best MAP value, so $N = 10, n = 9$ is selected in this experiment. (IR+QE).

- 3 Adopting the standard rating-based Weighted Slope-One Recommender algorithm to compute the prediction list, and combining this prediction results with IR results by combANZ (IR+WS1).

Besides the three baselines and the four methods we propose exploiting recommender techniques in the two and three-layer models in the PageRank component, we also carried out an experiment which utilizes a one-layer model for the PageRank algorithm in the integrated IR model. This method is referred to as (IR+PR). In this method each document was represented as a bag of terms, and we used cosine similarity to compute the affinity weight between documents in the selected topic category. The one-layer PageRank algorithm (introduced in Section 5.2.1) was used to output the document list ranked by importance. Finally, the IR output was re-ranked using the PageRank results using combANZ operator.

The experimental results for both baselines and the proposed approaches are shown in Table 5.1.

	MAP	P@5	P@10	P@20
IR	0.1225	0.1173	0.0947	0.0653
IR+QE	0.1498 (+22.29%)	0.1360 (+15.90%)	0.0880 (-7.07%)	0.0673 (+3.06%)
IR+WS1	0.1862 (+46.00%)	0.1729 (47.40+%)	0.1296 (+36.9%)	0.0857 (+31.2%)
IR+PR	0.2207* (+80.20%)	0.1947* (+65.90%)	0.1173* (+23.90%)	0.0913* (+39.80%)
Acos_PR	0.2410*† (+96.70%)	0.2013* (+71.58%)	0.1373*† (+44.89%)	0.0900* (+37.76%)
Cacos_PR	0.2526*† (+106.0%)	0.2005* (+70.85%)	0.1360*† (+41.53%)	0.1040*† (+40.92%)
Dev_PR	0.2454*† (+100%)	0.2027*† (+72.8%)	0.1387*† (+46.0%)	0.0851* (+30.5%)
CDev_PR	0.2593*† (+112%)	0.2035*† (+73.5%)	0.1384*† (+45.7%)	0.1040*† (+40.9%)

Table 5.1 Evaluation of all methods

From Table 5.1, we can see that the CDev_PR performs best among these methods. The IR+PR, Acos_PR, CAcos_PR, Dev_PR and CDev_PR methods achieve statistically significant improvements over standard IR results (marked with *). Acos_PR, CAcos_PR, Dev_PR and CDev_PR offer statistically significant (where the statistical significant evaluation with 95% confidence or 5% error) improvements over IR+PR marked with †. We observe from Table 5.3 that the CDev_PR method outperforms the other methods achieving significant improvements in MAP, P@5, P@10 and P@20 over both the results of IR and IR+PR.

In this experiment, we examined two recommender algorithms, the adjusted cosine similarity and the document deviation. From Table 5.1, it can be seen that the MAP of Acos_PR and Dev_PR, used in the two-layer models only has 0.0044 discrepancy, also the difference between the MAP of CAcos_PR and CDev_PR, using these two algorithms in the three-layer model is 0.0067 which is also very small. The explanation of this phenomenon is that both of these algorithms utilize the user ratings to compute the affinity weights for each document pair, which means that the affinity weight depends on this rating information. Since the same test collection is used for the experiments, the results are very similar.

The IR+PR run is the only one which uses the standard one-layer model in the PageRank algorithm, combined with the IR output. Table 5.1 clearly shows that our proposed four methods are significantly more effective than IR+PR. From the results shown in Table 5.3, we can see that the three-layer model runs have a relative good improvement than runs using two-layer model, where the MAP of CAcos_PR increases around 4.3% over Acos_PR and CDev_PR increases about 5.6% over Dev_PR run. We assume that if all the current queries could find their correct topic category, we would obtain even better results than this.

To sum up, the correlation between query and the topic category is a good factor to further improve the effectiveness of the integrated model over using the two-layer model in the PageRank algorithm for the recommender component.

5.4.4 Cross Validation Experiment

In order to further prove the effectiveness of this method, we conduct 10 fold cross validation experiments in this section. From Table 5.1, we observe that the CDev_PR method obtains the best results among all the experiments; thus in this experiment we choose Cdev_PR to conduct this validation evaluation.

Since we have tested this method once, we randomly select one test query from each topic (Note: if the topic category does not contain sufficient queries, the chosen query can be selected again in the cross validation) to run the experiment again, and repeat this process 9 times. Figure 5.7 shows the MAP arising from choosing different queries in each topic to conduct experiments using CDev_PR method.

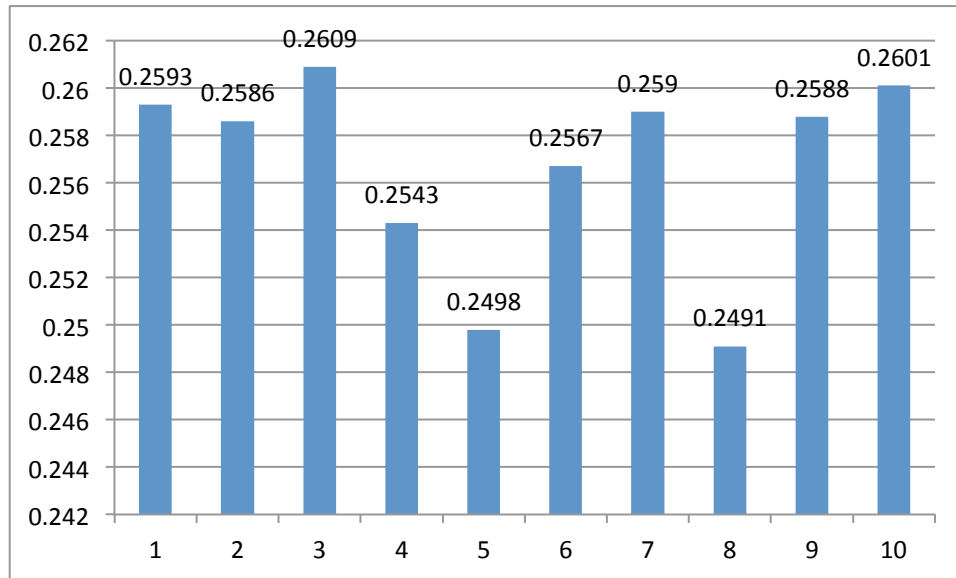


Figure 5.7 MAP of 10 cross validation experiments.

Figure 5.7 shows the result of these 10 cross validation experiments. We can see that in these 10 results, the lowest obtained MAP is 0.2491 and the largest is 0.2609. The average MAP for the 10 experiments is 0.2567. Obviously, this is still the best result compared to the others in Table 5.1. This result shows the effectiveness of using the recommender techniques in the PageRank algorithm for the recommender component in the integrated model.

From these results, we can conclude that utilizing recommender techniques to detect the features between documents in the cluster-based PageRank algorithm to re-rank IR output can improve on the results of standard IR algorithms.

This result indicates the answer for the following research question:

- What is the best method to exploit recommender techniques to improve the results of the integrated IR model?

In previous chapters, we exploited recommender algorithms directly to compute prediction results based on previous users search activities in the selected topic category, then merged these prediction results with IR output, obtaining positive results. In this chapter, instead of using a standard recommender component in the integrated model, we used the PageRank algorithm to generate a document importance ranking to replace the prediction results. Besides this, we also exploited the recommender techniques in the PageRank algorithm to compute document correlations based on previous users' search information, and combined the associated PageRank ranked output with IR output. The positive results obtained in this study indicate that this method performs better than using recommender algorithms directly. Based on all these observation, we can answer this research question that using recommender techniques in a PageRank component can improve

the effectiveness of the integrated model more than those which exploit them directly.

5.5 Summary

In this chapter, we proposed to adopt recommender techniques to detect correlations between documents in a cluster-based PageRank model. We also introduced a three-layer cluster-based PageRank model. Experiments again used the extended FIRE 2011 dataset, which tracks and records users' search behaviour, categorizes the users' search information into different topical categories. A link graph was built for each topic category and recommender techniques used to compute the affinity weight between documents. These correlations were applied to the PageRank algorithm, and finally an output list of the importance of documents in the selected topic category was generated. The document importance scores were utilized to re-rank the ranked output of a standard IR system. The proposed methods were compared with standard pseudo relevance feedback query expansion, a standard single-layer PageRank algorithm and recommender techniques applied directly to the IR system. Results showed that our new PageRank based methods perform more effectively than these baseline runs. We conclude that exploiting recommender features in the PageRank algorithm can improve over standard IR system results.

Two recommender techniques are investigated in this chapter: adjusted cosine similarity and Weighted Slope-One algorithm. Both of them compute the affinity weight between documents based on user ratings for these documents. The experimental results of using these two recommender techniques are quite similar, but both show that exploiting recommender techniques in the PageRank algorithm, and combining the PageRank results with standard IR results can improve retrieval results. We also observe that the relationship between query and

topic category, topic category and documents, and the relation between documents are key indicators for document relevance in the integrated model. Cross validation experiments are also conducted in this chapter, with the averaged results of these experiments showing the effectiveness of our proposed approach.

In next chapter, a method of using various query expansion methods in the IR component to improve the effectiveness of the integrated model is investigated. Besides this, we also examine the effectiveness of selecting varying numbers of topic categories for each current query and generate prediction rankings based on these selected topic categories, finally combining these results with the IR component output.

Chapter 6

Investigation of Query Expansion and Selection of Topic Categories

This chapter examines the performance of the integrated IR model using alternative settings to extend and reinforce the studies reported in the previous experiments. This chapter focuses on two topics: 1) investigation of the effectiveness of existing query expansion methods by exploiting users' search logs for the information retrieval (IR) component in the integrated model. 2) investigation of the effectiveness of selecting varying numbers of topic categories, instead of selecting the single most likely category as reported in the previous experiments.

6.1 Introduction

One of the key issues in IR is the matching of the user query terms against the contents of the documents in the available document collection. Users search requests typically bring two challenges to search engines:

- Users frequently describe their information needs using only a small number of words in their queries. Short queries submitted to search engines suffer from the problem that many important words or terms related to the user's information need

are not present in the queries. In this situation, it is a significant challenge for an IR system to use query to identify relevant documents.

- The words that users use to describe their information needs are often different from the index terms of the documents in the collection. This results in a word mismatch problem between the queries and the documents. As a consequence, in many cases, the documents returned by search engines are not relevant to the user's information need.

To address these two challenges in the integrated model, since we have search logs of previous users, we examine the use of a query expansion method based on utilizing this previous users' search information.

For the recommender component of the integrated IR model, in the previous chapter we investigated the use of alternative recommender algorithms to generate the prediction list, and also examined exploiting recommender techniques in the PageRank algorithm. The encouraging results of these experiments show the potential effectiveness of the integrated model. However, in these previous studies, we selected only a most likely topic category for each query to generate the prediction ranking list. When this category is selected correctly, the integrated model works more effectively than when an inappropriate category is selected. Selection of an inappropriate topic category can actually make the final results of the integrated model worse than the standard IR model. To attempt to address the weakness rising from incorrect topic category selection, in this chapter, we examine selection of more topic categories for each query to output multiple recommend prediction results. These lists are combined with the IR retrieval results to examine whether better results are obtained than when selecting only a single topic category.

Similar to Chapters 4 and 5, all experiments conducted in this chapter use the extended version of the FIRE 2011 personal information retrieval (PIR) data.

6.2 Query Expansion Using Collaborative Filtering Algorithm

Collaborative filtering (CF), overviewed in Section 2.2.2.2, is another approach to finding relevant information. CF is based on the assumption that a good way to identify relevant information is to find other people with similar interests and to recommend information that these similar users like. Note the similarity between this assumption and the one used for query expansion – in which a good way of finding terms to expand a query is to find documents that are similar to the query and to select the highest weighted terms from these documents to expand the query. Since CF algorithms have been proved to be effective for CF tasks, we could expect these algorithms to have value for query expansion. Indeed, Brandberg (2001) and Hoashi et al. (2001) demonstrate good performance in exploiting memory-based CF methods in query expansion to improve the effectiveness of the IR system results.

As described in previous chapters, in the extended FIRE 2011 test collection we recorded all previous users' search behaviour data and stored them in the dataset. The users' behaviour information included the documents that each user clicked and the rating they gave to each document. We observe that the data representation in our experiment is similar to that used in some CF methods, where user search logs can be viewed as a set of vectors. Each vector expresses a user, and the elements of each vector express the ratings of this user. This can be compared to the vector space model in IR, where each document and query is expressed as a vector of term weights within the retrieval system as illustrated in Figure 6.1.

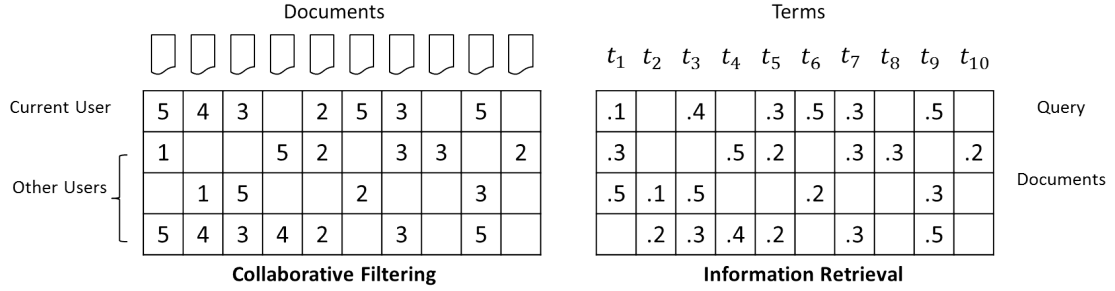


Figure 6.1 Example of similarity in data representation between Collaborative Filtering and Information Retrieval based on the Vector Space Model.

Based on this similarity, we examine use of CF methods for query expansion in the IR component of the integrated model. The expansion term selection is based on the recorded previous users' search behaviour and has two steps:

- 1 Utilize different CF methods to compute the prediction ranking for current query.
- 2 Use the top-ranked documents from the prediction ranking to extract terms to expand the current query.

In the next section, we compare this approach to query expansion based on CF with more standard IR query expansion methods.

6.3 Experimental Investigation of Query Expansion

In this section, we describe our experimental investigation of various methods for query expansion in the IR component of the integrated model. The results obtained for each approach are shown and analysed.

6.3.1 Using Pseudo Relevance Feedback for Query Expansion

As described in Section 2.1.1, query expansion via pseudo-relevance feedback (PRF) is based on the assumption that the top N documents in the initial retrieval list for a query are relevant, and

then adding n terms from these documents to the initial query. The Terrier IR toolkit provides various models for query expansion. We experimented with the default Bo1, introduced in Section 2.1.2, which is based on Bose-Einstein statistics and is similar to Rocchio (Amati, 2004). In Bo1, the informativeness $\omega(t)$ of a term t as previously introduced in Section 2.1.2 is given by Equation (6-1).

$$\omega(t) = tf_t \cdot \log_2 \frac{1 + P_n}{P_n} + \log_2(1 + P_n) \quad (6-1)$$

where tf_t is the frequency of the term t in the pseudo-relevant set (top N document set) and P_n is given by $F/|S|$. F is the term frequency of the query term in the whole collection and $|S|$ is the number of documents in the collection. The experiment reported here is the same as that described in Section 5.4.3. The Terrier Bo1 PRF method is used with the top 10 documents being feedbacks relevant and adding 9 additional terms to the current query, which were shown to obtain the best MAP value. This method is used as our baseline to compare with other query expansion methods introduced in the chapter.

The steps of using Bo1 PRF for query expansion are:

- 1 For each test query, pass the query to the Terrier search engine. Use the BM25 retrieval model ($k = 1.2, b = 0.75$) to obtain the initial retrieval ranking list for the query.
- 2 Take the top 10 documents from the initial retrieval ranking
- 3 Use Equation (6-1) to calculate each term weight in the 10 document set.
- 4 Select the $N = 9$ highest weight terms as the additional terms. Add them to the original current test query.
- 5 For the reformulated query, repeat the retrieval process (step 1) to obtain the retrieval results.
- 6 Evaluate the retrieval results using MAP and Precision at cut-off rank.

Results

The results for PRF query expansion method are shown in Table 6.1.

	MAP	P@5	P@10	P@20
Initial Retrieval Ranking	0.1225	0.1173	0.0947	0.0653
Using PRF Method Query Expansion	0.1498 (+22.29%)	0.1360 (+15.9%)	0.0880 (-7.07%)	0.0673 (+3.06%)

Table 6.1 MAP of Pseudo-relevance feedback method for query expansion

The table shows that, after query expansion, the results improve compared to the initial ranking. The MAP of PRF query expansion method results increase 22.29% and the precision at cut-off 5 increases from 0.1173 to 0.1360 (+15.9%).

6.3.2 Utilizing Recommender Component Output for Query Expansion

In this section, we investigate using the output of the recommender component as an alternative source of expansion terms for the current query. Two recommender methods for the generation of results by a recommender component were explored for query expansion: the popularity-focused rating-based filtering method (introduced and examined in Chapter 4) and the PageRank method (described in Chapter 5). We chose these two methods because their effectiveness has been examined in the previous chapters, and the results obtained were shown to outperform other existing algorithms in the integrated model. For the PageRank method, we utilized document deviation in the three-layer model because it achieved the best performance among the methods proposed in Chapter 5.

The process for extracting terms for query expansion using the recommender component was as follows:

Popularity-focused rating-based filtering method

- 1 For each training query, pass the query to the Terrier search engine. Use the BM25 retrieval model ($k = 1.2, b = 0.75$) to obtain the initial retrieval ranking list for the query.
- 2 As introduced in Chapter 4, use the popularity-focused rating-based filtering method to generate the prediction ranking for the current query. In brief, the steps are:
 - 1) Choose the best matching topic category for the query by comparing the similarity between the centroid representation of the current query and each topic category.
 - 2) Use the user search log data in the selected topic category with the popularity-focused rating-based filtering method to generate the prediction results.
- 3 The top M documents from the recommender component output ranking are selected as the source of expansion terms. Equation (6-1) is used to compute the term weight in order to select the highest ranked m terms for expansion. The selected m terms are added to the original current query.
- 4 The expanded query is passed to the Terrier search engine, and the BM25 retrieval model ($k = 1.2, b = 0.75$) again used to obtain re-ranked retrieval results.
- 5 The final retrieval results are evaluated using MAP to select the best parameter value, which was trained on the 123 training query set to decide how many top documents on the initial retrieval rank to select and how many expansion terms to choose.

Table 6.2 shows the MAP of using different values of M and m for the training set by using the prediction ranking obtained from the popularity-focused filtering method. From Figure 6.2., we can observe that when $M = 3, m = 8$ we obtain the best retrieval results for the expanded query on the training set.

	$M = 1$	$M = 2$	$M = 3$	$M = 4$	$M = 5$	$M = 6$
$m = 2$	0.1268	0.1276	0.1290	0.1325	0.1337	0.1321
$m = 3$	0.1272	0.1290	0.1286	0.1342	0.1365	0.1363
$m = 4$	0.1286	0.1288	0.1312	0.1396	0.1391	0.1389
$m = 5$	0.1296	0.1389	0.1320	0.1425	0.1436	0.1452
$m = 6$	0.1300	0.1389	0.1345	0.1469	0.1459	0.1460
$m = 7$	0.1305	0.1381	0.1368	0.1503	0.1498	0.1513
$m = 8$	0.1295	0.1313	0.1397	0.1564	0.1523	0.1537
$m = 9$	0.1300	0.1400	0.1406	0.1659	0.1561	0.1529
$m = 10$	0.1298	0.1404	0.1435	0.1662	0.1554	0.1532
$m = 15$	0.1196	0.1254	0.1302	0.1623	0.1502	0.1293
$m = 20$	0.1185	0.1248	0.1295	0.1597	0.1468	0.1225

Table 6.2 MAP value for varying the parameter values for M and m using popularity-focused filtering method for query expansion

- 6 Apply the parameter value $M = 3, m = 8$ for each query in the test set. For each test query, repeat steps 1 to 4 to obtain the retrieval results for the reformulated test query. Calculate the MAP and precision at cut-off rank for each test query.

PageRank method

The procedure for using PageRank method was used to generate a document importance ranking. This ranking was taken as the source of expansion terms to perform query expansion in a similar procedure to that when using the using popularity-focused rating-based filtering method.

- 1 For each training query, use the Terrier search engine with the BM25 retrieval model ($k = 1.2, b = 0.75$) to obtain the initial retrieval ranking list for the query.
- 2 Use the PageRank method with the document deviation algorithm to compute the affinity weight between documents in the three-layer model (as described in section 5.2.5) to output the document importance ranking list.

- 1) Choose the best matching topic category for the current query.
- 2) Build the three-layer link graph between the current query and the selected topic category.
- 3) Use the PageRank method to generate the document importance ranking list.
- 3 Select the top M documents from the recommender component output ranking as the source of expansion terms. Use Equation (6-1) to compute the term weights, in order to select the highest ranked m terms for expansion. The selected m terms are added to the original current query.
- 4 The expanded query is passed to the Terrier search engine, and the BM25 retrieval model ($k = 1.2, b = 0.75$) again used to obtain retrieval results.
- 5 The retrieval results are evaluated using MAP to select the best parameter values, of how many top documents from the initial retrieval rank are selected and how many expansion terms to choose.

	$M = 1$	$M = 2$	$M = 3$	$M = 4$	$M = 5$	$M = 6$
$m = 2$	0.1275	0.1284	0.1370	0.1334	0.1345	0.1328
$m = 3$	0.1279	0.1298	0.1366	0.1351	0.1373	0.1370
$m = 4$	0.1293	0.1396	0.1392	0.1405	0.1399	0.1396
$m = 5$	0.1303	0.1398	0.1400	0.1434	0.1444	0.1459
$m = 6$	0.1307	0.1397	0.1425	0.1478	0.1467	0.1467
$m = 7$	0.1312	0.1389	0.1448	0.1512	0.1506	0.1520
$m = 8$	0.1302	0.1321	0.1477	0.1573	0.1531	0.1544
$m = 9$	0.1307	0.1408	0.1486	0.1668	0.1569	0.1536
$m = 10$	0.1305	0.1412	0.1515	0.1671	0.1562	0.1539
$m = 15$	0.1203	0.1262	0.1382	0.1632	0.1510	0.1300
$m = 20$	0.1192	0.1256	0.1375	0.1606	0.1476	0.1232

Table 6.3 MAP values for varying the parameter values for M and m using PageRank for query expansion.

Table 6.3 shows the MAP values for varying numbers of M and m for the training set using document importance ranking obtained from the PageRank method. Table 6.3 indicates that when using the PageRank results for query expansion, the best retrieval results are achieved when $M = 4, m = 10$.

- 6 Based on the results shown in Table 6.3, for the PageRank method, the parameters were set to $M = 4, m = 10$, for the test query set.

The results for both methods on the test set are given in the following section

6.3.2.1 Results

The retrieval results for the query expansion using the recommender methods for PRF are shown in Table 6.4. This also shows the baseline IR results and using standard PRF for query expansion based on the IR component.

	MAP	P@5	P@10	P@20
Initial Retrieval Ranking	0.1225	0.1173	0.0947	0.0653
Using PRF Method Query Expansion	0.1498 (+22.29%)	0.1360 (+15.9%)	0.0880 (-7.07%)	0.0673 (+3.06%)
Using Popularity-Focused Filtering Output for Query Expansion	0.1662†* (+35.67%)	0.1495† (+27.45%)	0.0923 (-2.53%)	0.0682 (+4.44%)
Using PageRank Output for Query Expansion	0.1671 †* (+36.41%)	0.1500 †* (+27.88%)	0.0931 (-1.68%)	0.0679 (+3.98%)

Table 6.4 Evaluation of using recommender component output for query expansion.

† indicates significant improvement over the initial retrieval ranking. * Indicates significant improve over using PRF method for query expansion.

The results shown in Table 6.4 indicate that using the PageRank method to output results for query expansion performs slightly better than exploiting popularity-focused filtering to generate the prediction list for query expansion in the integrated model. Both of these methods improve over 35% compared to the baseline, and are shown to be more effective than PRF query expansion based on the baseline retrieval run.

6.3.3 Log-Based Query Expansion

This section describes our investigation of using query logs in a query expansion process. A log-based query expansion method was proposed by Cui et al. (2002), and was reviewed in Section 2.1.1.1. Compared to PRF methods which use the top-ranked documents for query expansion, the log-based method utilizes the documents clicked by the users to extract the expansion terms. The document clicks are more reliable indications than assuming the top-ranked documents from the initial retrieval ranking to be relevant. Because these clicked documents have been selected by human judgements as potentially relevant, not only are the clicked documents usually top-ranked, but also since they have been selected by real users, we can expect log-based query expansion method to be more robust and accurate than the PRF method.

6.3.3.1 Experimental Investigation

The process of utilizing the query log information to perform query expansion on a query Q was as follows:

- 1 Extract all query terms (eliminating stopwords) from query Q .
- 2 Find all documents related to any query term in query sessions.
- 3 For each document term in these documents, use Equation (6-1) to calculate its evidence of being selected as an expansion term according to the whole query.

$$CoWeight_Q(w_j^{(d)}) = \ln \left(\prod_{w_i^{(q)} \in Q} (P(w_j^{(d)} | w_i^{(q)}) + 1) \right) \quad (6-1)$$

- 4 Select m terms with the highest cohesion weight and formulate the new query Q' by adding these selected m terms into the original query.
- 5 Pass the new query Q' to Terrier search engine, using BM25 retrieval model ($k = 1.2, b = 0.75$) to obtain the retrieval ranking list.

- 6 Evaluate the retrieval results by MAP on the training dataset.

Figure 6.2. shows the MAP of adding varying numbers of expansion terms. The best results are received when $m = 12$, obtaining the best performed parameter value on the 123 training query set.

- 7 For each of the 27 test query set, repeat step 1 to 6 ($m = 12$) to obtain the retrieval results for each expanded test query set.
- 8 Finally, evaluate the MAP and precision at cut-off rank on the test set.

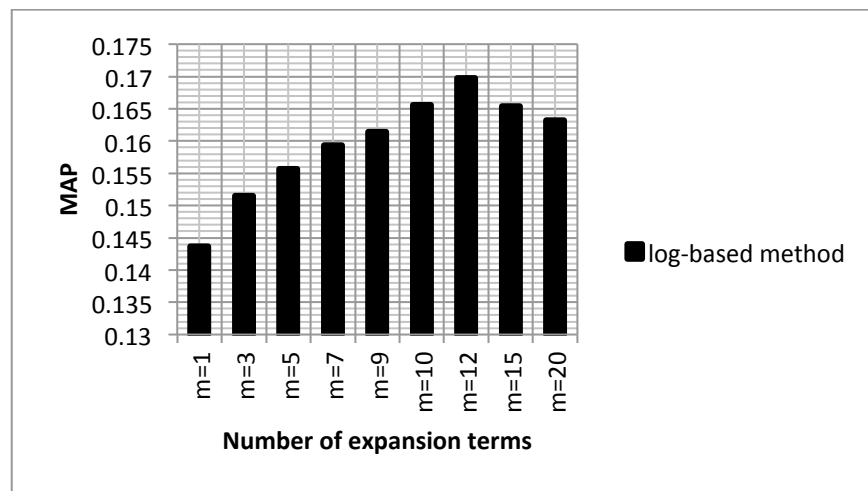


Figure 6.2 The MAP of vary parameter values for m using log-based method for query expansion.

6.3.3.2 Experimental Results

The results of using the log-based query expansion method with the test set are shown in Table 6.5. For comparison, the results of using query expansion methods investigated in previous sections methods are also shown in this table.

	MAP	P@5	P@10	P@20
Initial Retrieval Ranking	0.1225	0.1173	0.0947	0.0653
Using PRF Method Query Expansion	0.1498 (+22.29%)	0.1360 (+15.9%)	0.0880 (-7.07%)	0.0673 (+3.06%)
Using Popularity-Focused Filtering Output for Query Expansion	0.1662 [†] * (+35.67%)	0.1495 [†] (+27.45%)	0.0923 (-2.53%)	0.0682 (+4.44%)
Using PageRank Output for Query Expansion	0.1671 [†] * (+36.41%)	0.1500 [†] * (+27.88%)	0.0931* (-1.68%)	0.0679 (+3.98%)
Using Log-Based Query Expansion Method	0.1698[†]* (+37.88%)	0.1516[†]* (+30.35%)	0.0945* (-0.21%)	0.0678 (+3.97%)

Table 6.5 Evaluation of using log-based query expansion method, [†] represents significant improvement over the initial retrieval ranking, * indicates significant improvement over using PRF method for query expansion.

Table 6.5 shows the results of all query expansion methods we described in this section. It shows that using recommender algorithms, the PageRank method and the log-based method to perform query expansion have the significant improvement over the standard IR results. From the results we can see that extracting expansion terms from the output of recommender component for query expansion produces similar results to the log-based method in the integrated model. Overall, the log-based query expansion method's performance is a little better than the other two runs.

6.3.4 Summary for Query Expansion

In this section, we examined three methods to perform query expansion: a standard PRF method, two algorithms using the recommender output list to extract additional terms for the current query and exploiting user logs to calculate the correlation between query terms and document terms. From the results obtained, we observe that using user logs for the expansion achieved the best results.

Another observation is that the effectiveness of employing recommender component output for query expansion is close to the results of using user logs to perform query expansion. In this experiment, we chose two approaches, the popularity-focused method and the PageRank method,

to generate the recommender output, and used this for query expansion. From the results obtained in previous chapters we know that the PageRank method achieves better results than the popularity-focused method in the integrated model. Table 6.5 shows the results of the query expansion experiments. Again the PageRank method achieves better results than the popularity-focused approach. Based on this observation, we conclude that, in the integrated model, the better recommender prediction output is more suitable for query expansion experiment, which could be anticipated since the initial retrieval run is on average better.

6.4 Investigation of Varying the Number of Topic Categories Used in Recommender-Based Document Ranking

This section describes experiments examining selection of multiple topic categories in the recommender component of the integrated model for each query. A prediction ranking is produced for each of the selected topic categories, these predictions are then combined with the IR component output. The results obtained are compared with a selection of previous results using only the single best topic category for the current query for combination in the integrated model.

Experiments were again carried out using the extended version of the FIRE 2011 personal information retrieval (PIR) data.

The process of this experiment was similar to the experiments introduced in section 5.3.2. The procedure is as follows:

1 Obtain retrieval results for IR component:

The Terrier search engine, BM25 retrieval model ($k = 1.2, b = 0.75$) is used to obtain an initial retrieval ranking list for the query.

2 *Generate centroid documents for current query and each topic category:*

The procedure for generation of centroid document is the same as experiments described in section 4.3.2.2:

- Take the top 10 ranked documents from the initial retrieval list to generate the centroid document to be used as the representation of current query.
- For the topic category side, use the 5 highest frequency documents in each topic category to generate the centroid document to be used as the representation for its corresponding topic category.

The generated centroid documents was used as the representations for the current query. Each topic category model was used to categorize the current query to related topic categories.

3 *Categorizing:*

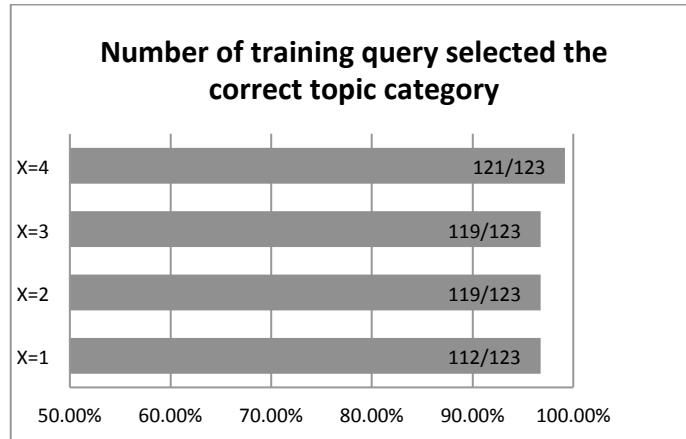


Figure 6.3 Number of training queries finding the correct topic category when selecting varying X topic categories for each training query.

Similar to the experiments described in Section 4.3.2 in the previous chapter, the cosine similarity method was used to categorize the current query to topic categories.

Figure 6.3 shows the number of queries which find the correct topic category when selecting a varying number of topic categories for the set of 123 training queries.

4 *Generate prediction:*

According to the experiments introduced in the query expansion section, the results of which are shown in Table 6.4, we observed that using the recommender algorithm with the PageRank method to generate the output for the recommender component, and combining it with the output of the IR component, achieved the best results in the integrated model. For this reason, we chose this method to conduct the experiment in this section. We use document deviation to compute the affinity weight between documents in the three-layer model for PageRank algorithm, as described in the Section 5.3.5, to generate the prediction for the selected X topic categories.

5 *Combination:*

In this step, in order to examine the performance of selecting a varying number of topic categories for the current query to create the output for recommender component in the integrated model, we chose three simple merging methods to combine the multiple ranked outputs from the recommender component with IR results:

- 1) CombANZ operator which combines IR retrieval ranking with the multi-prediction rankings directly. The reason for choosing the CombANZ operator for combination is that this method obtained the best results when combining two rankings in Chapter 4.
- 2) Linear combination without tune parameter for recommender component combination (Equation (6-2)). Then use CombANZ operator to combine the combined recommender output ranking with IR retrieval ranking.

$$Weight_d = P_{1_d} + P_{2_d} + P_{3_d} + \dots \quad (6-2)$$

where $Weight_d$ indicates the combined final prediction weight of document d , P_{i_d} denotes the prediction score of document d in prediction ranking i computed based on the users' search logs in topic category T_i . Then combine $Weight_d$ with the score of document d in IR results by CombANZ operator. (Linear+CombANZ)

- 3) Linear combination with weighting parameters to fuse the prediction rankings generated from each selected topic category, then use CombANZ operator to combine the combined recommender output ranking with IR retrieval ranking. When merging the prediction ranking lists, we give each list a parameter weighting according to the computed similarity between the current query and the topic categories which output these prediction lists, as shown by Equation 6-3.

$$Weight_d = \frac{Sim_{q,T_1}}{\sum_{i=1}^X Sim_{q,T_i}} \cdot P_{1_d} + \frac{Sim_{q,T_2}}{\sum_{i=1}^X Sim_{q,T_i}} \cdot P_{2_d} + \frac{Sim_{q,T_3}}{\sum_{i=1}^X Sim_{q,T_i}} \cdot P_{3_d} + \dots \quad (6-3)$$

where Sim_{q,T_i} denotes the similarity between the current query q and the selected topic category T_i . $\sum_{i=1}^X Sim_{q,T_i}$ is the sum of similarity between current query q and each selected topic category T_i and X is the number of selected topic categories.

After merging the prediction rankings, we use CombANZ to combine this integrated prediction ranking with the standard IR retrieval list. (WLinear+CombANZ)

6 Evaluation:

The retrieval effectiveness was again evaluated using MAP and precision at cut-off n . Table 6.6 shows the MAP of selecting X topic categories for the current query to generate

the prediction rankings, and combination of these multiple prediction results with IR results by using the selected fusion methods.

	$X = 1$	$X = 2$	$X = 3$	$X = 4$
CombANZ	0.2593	0.2609	0.2589	0.2580
Linear+CombANZ	0.2593	0.2598	0.2576	0.2555
WLinear+CombANZ	0.2593	0.2623	0.2601	0.2592

Table 6.6 MAP of alternative numbers of topic categories X to generate multiple prediction rankings, and combination with IR output.

From Table 6.6, we can observe that the WLinear+CombANZ merging method outperforms the method solely using CombANZ to merge all obtained ranking lists and using Linear+CombANZ to merge these obtained ranked lists. The best MAP occurs when $X = 2$, which means selecting two topic categories for the current query.

7 *Running Test Set:*

After running the previous steps on the training set, we applied the obtained parameter values on the test set.

Figure 6.4 shows the MAP of selecting different numbers of topic categories for each test query to generate the document importance rank for PageRank component, and combining these importance ranks using the WLinear+CombANZ fusion method, and finally combining the integrated PageRank component output with the IR output using the combANZ fusion operator. From this figure, we clearly observe than when $X = 2$, which means selecting two topic categories for each test query, the integrated model achieves the best results.

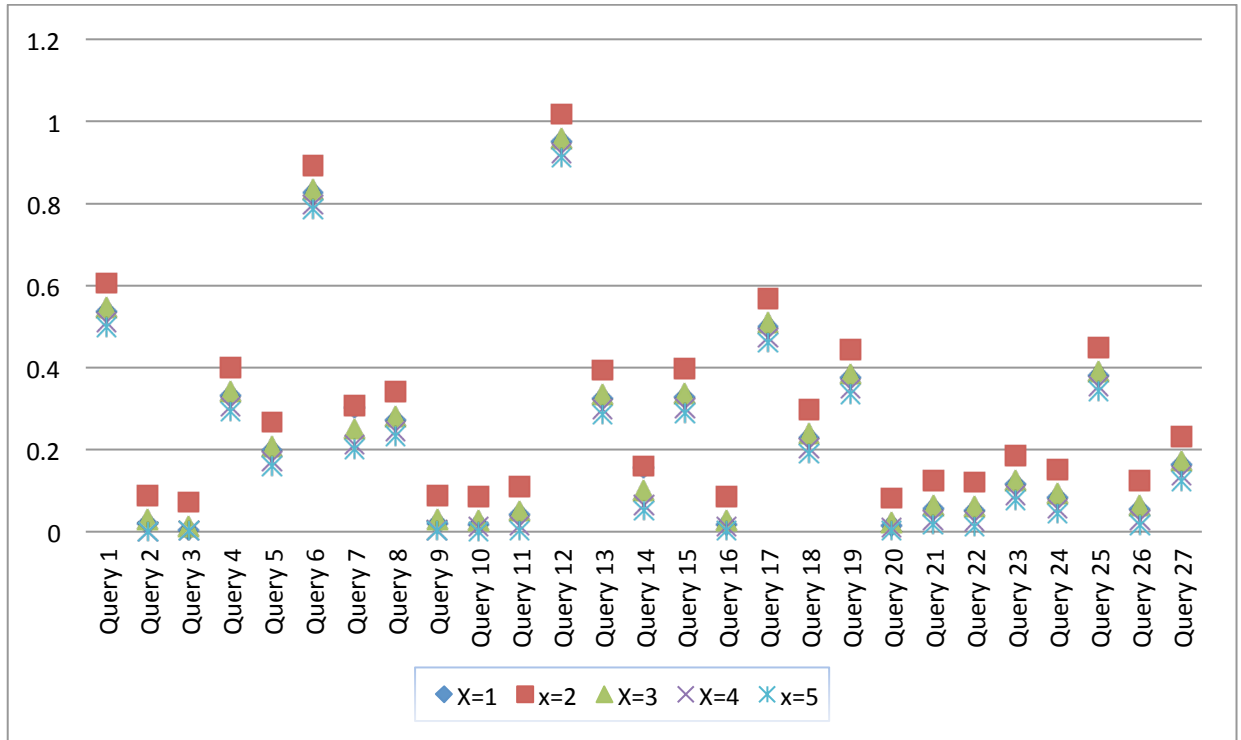


Figure 6.4 MAP of selecting topic categories for each test query to generate document importance rank, and combination with IR output.

For each of the 27 test queries, steps 1 to 6 were repeated. Firstly, we obtained the retrieval results using the Terrier search engine for the test query for IR component; Secondly, the same method was used to generate centroid documents for both current test query and topic categories. The top two best matching topic categories were selected for this query based on the similarity between the centroid representation of the query and the centroid representation of each topic category; Finally, the document deviation in PageRank method was used to compute the document importance rank, and this ranked list combined with the IR output using the three selected fusion methods.

	MAP	P@5	P@10	P@20
Initial Retrieval Ranking	0.1225	0.1173	0.0947	0.0653
Choose One Topic Category and Using PageRank Algorithm in Recommender Component	0.2593†	0.2035†	0.1384†	0.1040†
Choose Two Topic Category and Using PageRank Algorithm in Recommender Component	0.2623† (+1.15%)	0.2058† (+1.13%)	0.1396† (+0.87%)	0.1052† (+1.15%)

Table 6.7 MAP of comparison of initial retrieval results with selecting one topic category and selecting two topic categories for each test query and merging using the WLinear+CombANZ method.

† significant improvement over the initial retrieval ranking.

Table 6.7. shows that selecting the two most similar topic categories for each current query achieves slightly better results than selecting the only best matching topic category for combination in the integrated model, where the MAP of select two topic category increase 1.15% compare to only selecting one topic category for the current query. Choosing either one or two topic category for the PageRank in recommender component significantly improves over the standard IR system results.

Table 6.8 shows the results of combination of all best methods we used in this thesis, we use the log-based query expansion method in IR component to generate the IR output, and for recommender component, we selected the two highest match topic categories for each query and use the document deviation in the three-layer model in the PageRank algorithm to output the document importance rankings. The WLinear fusion method was used to fuse the two document importance ranking. Finally, the CombANZ operator was used to combine the output of the IR component and recommender components.

	MAP	P@5	P@10	P@20
Initial Retrieval Ranking	0.1225	0.1173	0.0947	0.0653
Choose One Topic Category and Use PageRank Algorithm in Recommender Component	0.2593†	0.2035†	0.1384†	0.1040†
Choose Two Topic Categories and Use PageRank Algorithm in Recommender Component	0.2623† (+1.15%)	0.2058 † (+1.13%)	0.1396† (+0.87%)	0.1052† (+1.15%)
Using Log-Based Query Expansion Method in IR Component & Choose Two Topic Category and Using PageRank Algorithm in Recommender Component	0.2698† (+4.05%)	0.2116† (+3.98%)	0.1408† (+1.73%)	0.1086† (+4.42%)

Table 6.8 The best results we achieved for the proposed integrated IR model in this thesis.

† presents that has significant improvement over the initial retrieval ranking.

6.4.1 Analysis of Selecting Multiple Topic Categories for Integrated Model

From the results obtained, we can draw the conclusion that selecting more than one topic category for each query to generate multi-prediction lists, and combining these results with IR output can achieve better results than only choosing one topic category in the integrated model. The results also reveal that the fusion method is important when combining the recommender component results and IR output. Since topic categories are ranked based on their similarity to the current query, treating the selected topic category equally when combining their prediction results may lead to the problem that non-relevant documents get high scores in the combined ranking. Table 6.6 shows this problem clearly. The CombANZ and Linear+CombANZ fusion methods obtain results worse than the WLinear+CombANZ method which uses weight parameters when combining multiple prediction rankings.

Another observation from the results obtained is that choosing more than one topic category only slightly increases the effectiveness compared to selecting only a most similar topic category for the current query. An analysis of this result showed that the main reason for this outcome is the dataset we used for the experiment. The extended FIRE 2011 dataset, which is composed of

Indian newspapers focuses on the news of India. Even when we extract 27 different topic categories, there are hidden correlations between some of them. Also some volunteers contribute very specific queries to the topic category they selected. All these reasons lead to the results that for both training set query and test set query, there is a significant chance that the top matching topic category is the correct one on the similarity descending ranking, where 112 out of 123 training queries find the correct topic category on the top rank; in the test set, the ratio is 23/27. Under this condition, although we obtained better results when choosing the top two ranked topic categories to generate predictions and combining these predictions with standard IR output, the results are only slightly better than when choosing only the top topic category. However, we have confidence that if using the same methods to conduct the experiment on a large and more realistic dataset which contains multiple topics, then selecting multiple topic categories for each query can achieve an improvement over only selecting one.

In the final section, we combined the output of the best methods we examined in this thesis, for the IR component. We use the log-based query expansion method to perform query expansion to output the IR results, and for each query, we choose the two highest matching topic categories. For the recommender component, we use the document deviation to compute the affinity weight between each document pair in the three-layer model and used this in the PageRank algorithm to output the document importance based on each selected topic category. Finally, we use the weighted linear combination to combine the two importance rankings, and finally combined this result with the IR output using the CombANZ operator. This combination produces our best results in this thesis. The MAP of this method is 0.2698, improves around 4.0% over the best results achieved in chapter 5.

In the next chapter, we seek to draw conclusions which from this thesis, answer the research questions we proposed in the beginning based on the experimental investigation we conducted in

the thesis, highlight the contributions of our research, and also propose possible future directions of this work.

Chapter 7

Conclusion and Future Direction

In this chapter, we return to the research question introduced at the beginning of the thesis, and develop conclusions based on the experimental investigations conducted in the previous chapters. The research question is answered based on our findings in the experiments. Finally, we propose potential future directions for this research.

7.1 Topic of the thesis

This thesis has presented a study examining the exploitation of community recommender models for enhancing IR, where the objective is to obtain better retrieval results than solely using an IR system. We propose an integrated model which combines the recommender and IR based components. The advantages and challenges of exploiting recommender algorithms to aid IR systems have been comprehensively discussed and explored. The study investigated three main aspects of the proposal:

- The effectiveness of the integrated IR model.
- Which recommender algorithms suit the integrated model.
- How to exploit recommender algorithms to optimize the integrated model.

This chapter summarizes the contributions of this study and outlines potential directions for future work.

7.2 Answer of Research Questions

The research questions have already been discussed separately in previous chapters. Here we revisit these questions and highlight their answers.

The research questions addressed in this thesis are as follows. For each question, various sub-questions need to be considered.

- *Can recommender techniques help an IR system to obtain better retrieval results?*

In this thesis, we exploit previous users' search behaviour to generate the prediction results for current query. This process is similar to the filter bubble¹⁹, we use the interests of similar users' activity to anticipate the current user's preferences. Unlike a filter bubble which eliminates items it thinks unrelated to current user's query, we generate all of these items in a ranked list based on their prediction score. These recommendations are used to re-rank items which are preferred by most similar interests users to the upper rank in the final combined result list. Finally this leads to a good precision result for the integrated model. So we can draw the conclusion that even though our integrated retrieval model has limitation (such as noisy sensitive), using the recommender techniques can still help to improve the retrieval results of standard IR system, or have the potential to further improve IR results.

¹⁹ A filter bubble is the intellectual isolation that can occur when websites make use of algorithms to selectively assume the information a user would want to see, and then give information to the user according to this assumption.

- *Can user logs be utilized to benefit the integrated IR model?*

According to the experiments, the answer for this question is positive. User logs can be used in two ways to improve the effectiveness of the integrated model. First, they can be used in the IR component to perform query expansion, in which similar preferences from the users' clicked items can help the current user to form a more precise query to present his/her information need. Second, user logs can be exploited by the recommender component to generate the prediction for current user, the clicked documents and the extracted dwell time are good indicators for predictions. The experimental results also show the effectiveness on various data collections. Based on these observations, we draw the conclusion that, utilizing the user logs is an effective way to improve the retrieval result, also for our integrated model.

- *Which recommender algorithms are most suitable in this application?*
 - *Which existing recommender algorithms benefit the integrated model most?*

Since each recommender algorithm has its advantage and drawbacks, it is hard to state which algorithm benefits the integrated model most, even after we conducted experiments to explore this, and examined the results that we obtained. We basically find that it mostly depends on the type of data collection we choose. The extended FIRE 2011 data collection we used in Chapter 4, the recorded user logs are a set of users' clicked documents and rating they give to those documents. It is clear that this data, the rating-based CF algorithm outperforms other recommender algorithms. However, since noise exists in the user logs, solely using the rating-based CF method may itself generate noise in the recommendations. To seek to address this problem content-based filtering method could be used to eliminate noisy documents from the recommendation process, since they will often appear as outliers in

terms of their content. So in our experiment, we can state that using a hybrid approach, which combines content-based filtering with rating-based CF to output a prediction result list and the output of a standard IR system can achieve the best results for the integrated model.

- *Do existing recommender algorithms provide suitable solutions in this application?*

According to the experiments in Chapter 4, we observe that although RSs and IR systems have similar broad objectives, using existing recommender algorithms directly in the integrated model is not the best way to optimize its effectiveness. Recommender algorithms are designed to generate novel and popular items to users, however these items may not be relevant to the current query. In order to address this problem, involving some IR features into existing recommender algorithms to make them better suited for retrieval applications is a better approach for the integrated IR model.

- *What is the most effective method to exploit recommender techniques to optimize the results of the integrated IR model?*

We examined two different methods of exploiting recommender techniques in the integrated model. One is to use them directly to output a prediction ranking of documents, the other is to use them in the PageRank algorithm to compute the correlation between documents. Results show that the latter method achieves better results than first one. The conclusion drawn from our experiment is that using the features of recommender algorithms in IR method probably is the most effective method to optimize the results of the integrated retrieval model.

7.3 Contribution of the thesis

In this thesis, we proposed an IR integrated model which combines RSs with an IR system to improve the effectiveness of standard IR systems. We investigated the effectiveness of exploiting various existing recommender algorithms to benefit IR system results, and observed the weakness of using existing algorithms in the integrated model. A method of involving IR system focused features in the rating-based CF method was then proposed to improve the retrieval results. Besides exploiting recommender techniques directly, we also proposed an approach that utilizes recommender techniques in the PageRank algorithm to further improve the effectiveness of the integrated model. In summary, this thesis makes contributions as follows:

- 1 Proposed a framework of an integrated retrieval model which combines RSs with IR system to improve the standard IR retrieval results based on previous user logs.
- 2 Examined the performance of various existing recommender algorithms in the integrated model, and determined weaknesses of existing models. Investigated a novel method of adding IR features in a recommender algorithm and observed this method can improve the performance of the recommender component in the integrated model.
- 3 Investigated different methods of exploiting recommender techniques, directly and indirectly. Observed that using recommender algorithms in a PageRank component is an effective way to improve the performance of the integrated model.

In general, the experiments described in this thesis address the hypothesis we made in the first chapter, i.e. whether RSs can improve IR system? From the our invesigation of this question, the answer for this is positive. Results obtained support the conclusion that combining recommender techniques with IR systems can significantly improve IR system retrieval results.

7.4 Possible Future Directions

We examined the integrated model on simulated and user created test collections, and proposed algorithms to improve the effectiveness of the model. We obtained positive results for these proposed methods. However, each of our proposed models or algorithms work with some limitations. In an era when large amounts of data are widely available, scalability becomes an important measure to gauge the merit of a method. For the integrated model, scalability is an important issue which would need to be explored in further study.

One important constraint for our model is the small test collection which may not be sufficiently representative enough of a larger dataset. Although we collected the search logs from real volunteer users, this test collection does not cover some real world cases, such as the situation where the topic categories have poor coverage of the user queries. In this case, cluster methods could be used to automatically create new topic categories. In future work, we intend to explore the integrated model on large amounts of training data to address these challenges to make the integrated model more robust.

Another important issue of this work is the noise in the feedback data which is collected from users, which is a big challenge for RS systems. We did not address this problem in this work. In future work, we would do deeper analysis on users' feedback data to find out the threshold to decide what types of data can be assumed to be noise documents to be discarded. Noisy sensitivity is a weakness of our integrated method, which we would need to explore in future work to attempt to find a method to decrease the impact of noise in the integrated model.

Another problem of the model is the fusion method for combining the IR output with the recommender component outcome. In this thesis, we combine the outputs of two components directly. However, we observe that the final results are sensitive to the combination parameters.

In future work, we intend to investigate fusion methods which can tune the combination parameters automatically based on confidence in the two components for a specific query.

Another potential area of further work is to investigate other approaches to exploiting recommender techniques to aid IR systems in the integrated model. We have examined direct use of recommender algorithms in the recommender component to compute the prediction list for the current query and combining this prediction outcome with IR output, and also explored recommender techniques in a PageRank model to generate the document importance list and combining it with IR output. Both approaches obtained positive results, the latter one better than first one. In future work, we plan to investigate other methods of exploiting recommender techniques in the integrated model to improve its effectiveness.

The investigations described in this thesis have focused on retrieval for text documents. Increases on archives of multimedia content including speech, image, video and music are creating needs for effective retrieval systems for content of these types. One of the main challenges of IR for multimedia content is the need to analyze the content to find out what it contains. In this situation recommender systems based on collaborative methods are attractive since they rely only user behaviour without needing to analyze the content itself. At the same, the difficulties of analyzing the content and specifying search queries based on content, mean that often search of this content relies on textual metadata assigned to the content, either from analysis of the content or manually assigned. In either, content-based retrieval of multimedia content often performs rather poorly, with either or both of low precision or low recall. Combination of content-based retrieval with CF recommender methods would thus appear to have considerable potential and forms an appealing additional area for further work.

Bibliography

- Aggarwal, C. C., Wolf, J. L., Wu, K.L., and Yu, P. S., Horting hatches an egg: a new graph-theoretic approach to collaborative filtering. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '99)*. pages 201-212, ACM, New York, USA. (1999)
- Agichtein E., Brill E. and Dumais S., Improving web search ranking by incorporating user behaviour information. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2006)*. pages 19-26, New York, USA. (2006)
- Alzghool M. and Inkpen D., A Novel Class-Based Data Fusion Technique for Information Retrieval. In *Journal of Emerging Technologies in Web Intelligence*, vol 2, no. 3, pages 160-166. (2010)
- Amati G., Probabilistic Model for Information Retrieval Based on Divergence from Randomness. PhD thesis, Department of Computing Science, University of Glasgow. (2003)
- Amati G., Carpineto C. and Romano G., Query Difficulty, Robustness and Selective Application of Query Expansion. In *Proceedings of the 26th European Conference on IR Research (ECIR 2004)*, Sunderland, U.K., pages 127-137. (2004)
- Armstrong J.S., Principles of Forecasting--A Handbook for Researchers and Practitioners. Series: International Series in Operations Research & Management Science. vol 30, XII, page 849. (2001)
- Aronson S.R. and Rindflesch T.C., Query Expansion Using the UMLS Metathesaurus. In *Proceedings of AMIA Annu Fall*. pages 485-489. (1997)

- Arora R. and Ravindran B., Latent Dirichlet Allocation Based Multi-Document Summarization. In *Proceedings of the Second Workshop on Analytics for Noisy Unstructured Text Data (AND'08)*. pages 91-97. New York, USA. (2008)
- Baeza-Yates R., Ribeiro B.A.N. and Ribeiro-Neto B., Modern Information Retrieval. ACM Press. (1999)
- Baeza-Yates, J., and Ribeiro-Neto, B., Modern Information Retrieval: The Concepts and Technology Behind Search. ACM Press. (2010)
- Balabanovic M., Shoham Y. Fab, Content-Based, Collaborative Recommendation. In *Journal of Commun. ACM*, vol 40, no. 3, pages 66-72. (1997)
- Basilico J., and Hofmann T., Unifying Collaborative and Content-Based Filtering. In *Proceedings of the 21st International Conference on Machine Learning (ICML'04)*. page 9, New York, USA. (2004)
- Basu C., Hirsh H. and Cohen W., Recommendation as Classification: Using Social and Content-Based Information in Recommendation. In *Proceedings of 15th National/10th Conference on Artificial Intelligence/ Innovative Application of Artificial Intelligence Recommender System Workshop, Technical Report WS-98-98, AAAI Press*. pages 714-720, CA, USA. (1998)
- Bates M.J., The Design of Browsing and Berrypicking Techniques for the Online Search Interface. *Online Review*, vol. 13, no, 5, pages 407-424. (1989)
- Bates M.J. and Williams M.E., Search Techniques. *Annual Review of Information Science and Technology*, vol 16, pages 139-169.(1981)
- Beeferman D. and Berger A., Agglomerative Clustering of a Search Engine Query Log. In *Proceedings of the 6th ACM SIGKEE International Conference on Knowledge Discovery and Data Mining*. pages 407-416. New York, USA. (2000)

- Belkin N.J. and Croft W.B., Information Filtering and Information Retrieval: Two sides of the same coin? *Communications of the ACM - Special Issue on Information Filtering*, vol 35, no 12, pages 29-38 (1992)
- Bellogin A. and Wang J., Text Retrieval Methods for Items Ranking in Collaborative Filtering. In *Proceedings of ECIR 2011: The 33rd European Conference on Information Retrieval*. pages 301-306, Heidelberg. (2011)
- Bhogal J., Macfarlane A. and Smith P., A Review of Ontology Based Query Expansion. *International Journal of Information Processing & Management*. vol 43, no 4, pages 866-886. (2007)
- Billsus D. and Pazzani M., Learning Collaborative Information Filtering. In *Proceedings of the 5th International Conference on Machine Learning (ICML'98)*. pages 46-54, San Francisco, USA. (1998)
- Brandberg G., Query Expansion Using Collaborative Filtering Algorithm. Master Thesis in Computing Science of Uppsala University, Sweden. (2001)
- Breese J.S., Heckerman D. and Kadie C., Empirical Analysis of Predictive Algorithms for Collaborative Filtering. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI'98)*. pages 43-52, CA, USA. (1998)
- Brin S. and Larry P., The Anatomy of a Large-Scale Hypertextual Web Search Engine. In *Proceeding of the 7th International World-Wide Web Conference (WWW'98)*. pages 107-117, The Netherlands. (1998)
- Brusilovsky P. and Tasso C., Preface to Special Issue on User Modeling for Web Information Retrieval. *Journal of User Modeling and User-Adapted Interaction*, vol 14, no 2-3, pages 147-157. (2004)

- Buckley C., Salton G. and Allan J., Automatic Retrieval with Locality Information Using Smart, In *Proceedings of the 1st Text Retrieval Conference (TREC-1)*, pages 59-72. (1992)
- Buckley C., Mitra M., Walz J., and Cardie C., Using Clustering and Superconcepts within Smart. In *Proceedings of the 6th Text Retrieval Conference (TREC-6)*, pages 107-124. (1998)
- Buckley C., Salton G., Allan J., and Singhal A., Automatic Query Expansion Using SMART. In *Proceedings of the 3rd Text Retrieval Conference (TREC-6)*, pages 69-80. (1994)
- Buckley, C., Dimmick, D., Soboroff, I., and E. Voorhees. Bias and the Limits of Pooling. In *Proceedings of the 29th ACM International SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2006)*, Pages 619-620, Seattle, USA. (2006)
- Cacheda F., Carneiro V., Fernandez D. and Formoso V., Comparison of Collaborative Filtering Algorithms: Limitaions of Current Techniques and Proposals for Scalable, High-Performance Recommender Systems. *ACM Transactions on Web (TWEB)* vol 5, no 1, Article No. 2. (2011)
- Cao G.H., Nie J.Y., Gao J.F. and Robertson S., Selecting Good Expansion Terms for Pseudo-Relevance Feedback. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2008)*. pages 243-250. Singapore. (2008)
- Canny J., Collaborative Filtering with Privacy via factor analysis. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2002)* . pages 238-245. Tampere, Finland. (2002)
- Carpineto C., Romano G., and Bigi B., An Information-Theoretic Approach to Automatic Query Expansion. *ACM Trans. Information Systems*. vol. 19, no. 1, pages 1-27. (2001)

- Chee S.H.S., Han J.W. and Wang K., Rectree: An Efficient Collaborative Filtering Method. In *Proceedings of the 3rd International Conference on Data Warehousing and Knowledge Discovery (DaWaK'01)*. pages 141-151. Munich, Germany. (2001)
- Chen S. and Goodman J., An Empirical Study of Smoothing Techniques for Language Modeling. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL'96)*. pages 310-318. California, USA. (1996)
- Chirita P.A., Nejdl W. and Zamfir C., Preventing Shilling Attacks in Online Recommender Systems. In *Proceedings of the 7th Annual ACM International Workshop on Web Information and Data Management (WIDM'05)*. pages 67-74. Bremen, Germany. (2005)
- Choi F.Y.Y., Advances in domain independent linear text segmentation. In *Proceedings of the 1st North American Chapter of the Association for Computational Linguistics conference (NAACL'00)*, pages 26–33, Seattle, WA, USA (2000)
- Claypool M., Gokhale A., Miranda T., Murnikov P., Netes D. and Sartin M., Combining Content-Based and Collaborative Filters in an Online Newspaper. In *Proceedings of ACM SIGIR'99 Workshop Recommender Systems: Algorithms and Evaluation*. Berkeley, California, USA. (1999)
- Cohen E, Lewis D.D., Approximating Matrix Multiplication for Pattern Recognition Tasks. In *Proceedings of the 8th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'97)*. pages 682-691. New Orleans, Louisiana, USA. (1999)
- Collins-Thompson K. and Callan J., Query expansion using random walk models. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management (CIKM 2005)*. pages 704-711. Bremen, Germany. (2005)
- Cöster R, Svensson M., Inverted File Search Algorithms for Collaborative Filtering. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and*

- Development in Information Retrieval (SIGIR 2002)*. pages 246-252. Tampere, Finland. (2002)
- Costa A. and Roda F. Recommender Systems by means of Information Retrieval. In *Proceedings of the International Conference on Web Intelligence, Mining and Semantics (WIMS 2011)*. Article No. 57. Sogndal, Norway. (2011)
 - Croft W.B., Knowledge Based and Statistical approaches to text retrieval. *IEEE Expert: Intelligent Systems and Their Applications*. vol 8, no 2, pages 8-12. (1993)
 - Croft W.B. and Harper D.J., Using Probabilistic Models of Document Retrieval without Relevance Information. *Document Retrieval Systems*. pages 161-171. (1979)
 - Crouch C.J. and Yang B., Experiments in Automatic Statistical Thesaurus Construction, In *Proceedings of ACM-SIGIR Conference Research and Development in Information Retrieval (SIGIR 1992)*. pages 77-88. Copenhagen, Denmark. (1992)
 - Cui H., Wen J.R., Nie J.Y. and Ma W.Y. Query Expansion by Mining User Logs. *IEEE Transaction on Knowledge and Engineering*. vol 15, no 4, pages 829-839. (2002)
 - Dang V. and Croft B.W., Query reformulation using anchor text. In *Proceedings of the 3rd ACM International Conference on Web Search and Data Mining (WSDM'10)*. pages 41-50. New York City, USA. (2010)
 - Deerwester S., Dumai S.T., Furnas G.W., Landauer T.K. and Harshman R., Indexing by Latent Semantic Analysis. *Journal of Information Science and Technology*, vol. 41, no. 6, pages 391-407. (1990)
 - Delgado J. and Ishii N., Memory-Based Weighted-Majority Prediction for Recommender Systems. In *Proceedings of the 22nd Annual International ACM SIGIR Conference Workshop on Recommender System Algorithms and Evaluation*. Berkeley, California, USA. (1999)

- Efthimiadis E. and Biron P., UCLA-Okapi at TREC-2: Query Expansion Experiments. In *Proceedings of 2nd Text Retrieval Conference (TREC-2)*. pages 278-290. Gaithersburg, Maryland. (1994)
- Efthimiadis E.N., Query Expansion. In *Annual Review of Information Systems and Technology (ARIST)*. vol. 31, pages 121-187 (1996)
- Eichmann D. and Ruiz M.E., Cross-Language Information Retrieval with the UMLS Metathesaurus. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1998)*. pages 72-80. Melbourne, Australia. (1998)
- Eui-Hone H. and George K., Centroid-Based Item Classification: Analysis & Experimental Results. In *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD '00)*. pages 424-431. Lyon, France. (2000)
- Evans D. and Lefferts R., Design and Evaluation of the CLARIT-TREC-2 System. In *Proceedings of the Second Text Retrieval Conference. (TREC-2)*. page 137. Gaithersburg, Maryland, USA. (1994)
- Foltz P.W. and Dumais S.T., Personalized Information Delivery: An Analysis of Information Filtering Methods. *Communications of the ACM - Special Issue on Information Filtering*. vol 35, no 12, pages 51-60. (1992)
- Funk S., Netflix Update: Try This at Home. <http://sifter.org/simon/journal/20061211.html> (2006)
- Ganguly D., Leveling J., Curtis K. Li,W. and Jones G.J.F., Overview of the Personalized and Collaborative Information Retrieval (PIR) Track at FIRE-2011. In *Forum for Information Retrieval Evaluation (FIRE) 2011 Workshop*, Bombay, India. (2011)

- Ganguly D., Leveling J. and Jones G.J.F., Simulation of Within-Session Query Variations using a Text Segmentation Approach. In *Proceeding of CLEF 2011 Conference on Multilingual and Multimodal Information Access Evaluation*, pages 89-94. Amsterdam, Netherland. (2011)
- Gauch S., Speretta M., Chandramouli A. and Micarelli A. User Profiles for Personalized Information Access. *Adaptive Web*. pages 54-89. (2007)
- Gediminas A. and Alexander T., Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Transactions on Knowledge and Data Engineering*. vol 17, no 6, pages 734-749. (2005)
- Goldberg K., Roeder T., Gupta D. and Perkins C., Eigentaste: A Constant Time Collaborative Filtering Algorithm. *Journal of Information Retrieval*. vol 4, no 2, pages 133-151. (1999)
- Hardtke D., Wertheim M. and Cramer M., Demonstration of Improved Search Result Relevancy Using Real-Time Implicit Relevance Feedback. In *SIGIR 2009 Workshop on Understanding the Users*. Boston, MA, U.S.A. (2009)
- Harter S.P., Online Information Retrieval: Concepts, Principles and Techniques. *Journal of the American Society for Information Science*. vol 38, no 4, page 316. (1986)
- Hearst M.A., TextTiling: Segmenting Text into Multi-paragraph Subtopic Passages. *Journal of Computational Linguistics*, vol 23, pages 33-64. (1997)
- Herlocker J.L., Konstan J.A., Terveen L.G. and Riedl J.T., Evaluation Collaborative Filtering Recommender Systems. *ACM Transactions on Information Systems (TOIS)*. vol 22, no 1, pages 5-53. (2004)
- Herlocker J.L., Konstan J.A., Borchers A. and Riedl J.T., An Aglroithmic Framework for Performing Collaborative Filtering. In *Proceedings of the 22nd Annual International ACM*

- SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1999).* pages 230-237. Berkeley, California, USA. (1999)
- Hiemstra D., Using Language Models for Information Retrieval. PhD thesis, Center of Telematics and Information Technology, University of Twente. (2001)
 - Hiemstra D. and Kraaij W., Twenty-One at TREC-7: Ad-Hoc and Cross-Language Track. In *Proceedings of the 7th Text Retrieval Conference TREC-7*. Gaithersburg, MD, USA (1998)
 - Hoashi K., Matsumoto K., Inoue N. and Hashimoto K., Query Expansion Based on Predictive Algorithms for Collaborative Filtering. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and Development in Information Retrieval*. pages 414-415. New Orleans, Louisiana, USA. (2001)
 - Hofmann T., Latent Semantic Models for Collaborative Filtering. *ACM Transaction Information System*. pages 89-115. (2004)
 - Hu Y., Koren Y., Volinsky C., Collaborative Filtering for Implicit Feedback Datasets. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM'08)*.. pages 263-272. Pisa, Italy. (2008)
 - Huang Z., Chen H. and Zeng D., Applying Associative Retrieval Techniques to Alleviate the Sparsity Problem in Collaborative Filtering. *ACM Transactions on Information System (TOIS)*. vol 22, no 1, pages 116-142. (2004)
 - Jansen B. J., Spink A., Bateman J. and Saracevic T. Real Life Information Retrieval: A Study of User Queries on the Web. *ACM SIGIR Forum*. vol 32, no 1, pages 5-17. (1998)
 - Jeon H., Kim T. and Choi J., Adaptive User Profiling for Personalized Information Retrieval. In *Proceedings of 3rd International Conference on Convergence and Hybrid Information Technology (ICCIT'08)*. vol 2, pages 836-841. Busan. (2008)

- Jing Y. and Croft W.B., An Association Thesaurus for Information Retrieval. In *Proceedings of the 4th International Conference RIAO 1994*. pages 146-160. New York, USA. (1994)
- Kohrs A. and Merialdo B., Clustering for Collaborative Filtering Application. In *Proceedings of the International Conference on Computational Intelligence for Modelling, Control and Automation (CIMCA '99)*. Vienna, Austria. (1999)
- Koren Y., Bell R., and Volinsky C., Matrix Factorization Techniques for Recommender Systems. *IEEE Computer*. vol 42, no 8, pages 30-37. (2009)
- Kurland O. and Lee L., PageRank without Hyperlinks: Structural Re-ranking Using Links Induced by Language Models. In *Proceedings of the Twenty-Eighth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'05)*. pages 306-313. Brazil. (2005)
- Kurland O. and Lee L., Respect my Authority! HITS without Hyperlinks, Utilizing Cluster-Based Language Models. In *Proceedings of 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'06)*. pages 83-90. Seattle, USA. (2006)
- Labadié A. and Prince V., Lexical and Semantic Methods in Inner Text Topic Segmentation: A Comparison between C99 and Transeg. In *Proceedings of Natural Language and Data Bases (NLDB'08)*. pages 347-349. London, UK. (2008)
- Lam S. and Riedl J., Shilling Recommender Systems for Fun and Profit. In *Proceedings of the 13th International Conference on World Wide Web (WWW'04)*. pages 393-402. New York, USA. (2004)
- Lam-Adesina A.M. and Jones G.J.F., Applying Summarization Techniques for Term Selection in Relevance Feedback, In *Proceedings of the 24th Annual International ACM*

- SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2001)*, pages 1-9. New Orleans, USA. (2001)
- Lang K., Newsweeder: Learning to Filter Netnews. In *Proceedings of the 12th International Conference on Machine Learning (ICML 1995)*. Pages 331-339. California, USA. (1995)
 - Larry P., PageRank: Bridging Order to the Web. Stanford Digital Library Project, Talk. (1997) (Archived 2002)
 - Larry P., Brin S., Motwani R. and Winograd T., The PageRank Citation Ranking: Bringing Order to The Web. *Technical Report*. Stanford Digital Libraries. (1998)
 - Lee J.H., Combining Multiple Evidence from Different Properties of Weighting Schemes. In *Proceedings of the 18th annual international ACM SIGIR conferment on Research and development in information retrieval*. pages 180-188. Seattle, USA. (1995)
 - Lemire D. and Maclachlan A., Slope One Predictors for Online Rating-Based Collaborative Filtering. In *Proceedings of SIAM 2005: International Conference on Data Mining*. pages 471-475. Newport Beach, California, USA. (2005)
 - Lesk M.E., Word-Word Associations in Document Retrieval Systems. *American Documentation*. vol. 20, no 1, pages 27-38. (1969)
 - Li J., Zaïane O.R. and Zaïane J., Combining Usage, Content and Structure Data to Improve Web Site Recommendation. In *Proceedings of the Fifth International Conference Electronic Commerce and Web Technologies (EC-Web'04)*. pages 305-315. Zaragoza, Spain. (2004)
 - Li W., Ganguly, D. and Jones G.J.F., Enhanced Information Retrieval Using Domain-Specific Recommender Model. In *Proceedings of the 3rd International Conference on Advances in Information Retrieval Theory (ICTIR'11)*. pages 201-212. Bertinoro, Italy. (2011)

- Li W. and Jones G.J.F., Utilizing Recommender Algorithms for Enhanced Information Retrieval. *In Proceedings of the 10th Conference on Open Research Areas in Information Retrieval (OAIR'13)*. pages 141-144. Lisbon, Portugal. (2013)
- Li W. and Jones G.J.F., Enhanced Information Retrieval by Exploiting Recommender Techniques in Cluster-Based Link Analysis. *In Proceedings of the 4th International Conference on Advances in Information Retrieval Theory (ICTIR'13)*. page 12. Copenhagen, Denmark. (2013)
- Library of Congress Subject Headings: <http://id.loc.gov/authorities/subjects.html>
- Lilien G.L., Kotler P. and Moorthy K.S., Marketing Model. Prentice Hall of Indian. (1992)
- Linden G., Smith B. and York J, Amazon.com Recommendations: Item-to-Item Collaborative Filtering. *IEEE Internet Computing*. vol 7, no 1, pages 76-80. (2003)
- Liu C., Liu J.J., Belkin N., Cole M. and Gwizdka J., Using Dwell Time as an Implicit Measure of Usefulness in Different Task Types. *In Proceedings of the American Society for Information Science and Technology (ASIS&T)*. vol 48, no 1, pages 1-4. New Orleans, Louisiana, USA. (2011)
- Liu T.Y. and Ma W.Y., Webpage Importance Analysis Using Conditional Markov Random Walk. *In Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence (WI'05)*. pages 515-521. Compiègne, France. (2005)
- Liu Y., Gao B., Liu T.Y., Zhang Y., Ma Z.m., He S.Y. and Li H.: BrowseRank: Letting Web Users Vote for Page Importance. *In Proceedings of Thirty-First Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'08)*. pages 451-458. Singapore. (2008)
- Lops P., Gemmis M. and Semeraro G., Content-Based Recommender Systems: State of the Art and Trends. *Recommender Systems Handbook, chapter 3*. pages 73-105. (2011)

- Lovin J.B., Development of a Stemming Algorithm. In *Proceedings of Mechanical Translations and Computational Linguistics*. Vol 11, Nos 1 and 2. (1968)
- Lu, M. Ayoub, and J. Dong, Ad Hoc Experiments Using EUREK. In *Proceedings of the 5th Text REtrieval Conference (TREC-5)*. pages 229-240. (1997)
- Lu F., Li X. and Liu Q. T., Research on Personalized E-Learning System Using Fuzzy Set Based Clustering Algorithm. *Lecture Notes in Computer Science*. Vol 4489, pages 587-590. Springer Berlin, Heidelberg. (2007)
- Lu K. and Mu X.M., Query Expansion Using UMLS Tools for Health Information Retrieval. In *Proceedings of the American Society for Information Science*. vol 46, no 1, pages 1-16. (2009)
- Lu Y., Fang H. and Zhai C.X., An Empirical Study of Gene Synonym Query Expansion in Biomedical Information Retrieval. *Journal of Information Retrieval*. Vol. 12, pages 51-68. (2009)
- Luhn H., A Statistical Approach to Mechanised Encoding and Searching of Literary Information. *IBM Journal of Research and Development* 1. pages 309-317. (1957)
- Lv, Y. and Zhai, C., Positional relevance model for pseudo-relevance feedback. In *Proceedings of the 33rd international ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2010)*. pages 579-586. Geneva, Switzerland. (2010)
- Macdonald, C., He, B., Plachouras, V., & Ounis, I. University of Glasgow at TREC 2005: Experiments in terabyte and enterprise tracks with terrier. In *Proceedings of the 14th Text REtrieval Conference (TREC 2005)*. Gaithersburg, MD. (2005)
- Magdy W., Toward Higher Effectiveness for Recall-Oriented Information Retrieval: A Patent Retrieval Case Study. PhD thesis, School of Computing, Dublin City University. (2011)

- Manning C.D., Raghavan P. and Schütze H., Introduction to Information Retrieval. Cambridge University Press. (2008)
- Marlin B., Collaborative Filtering: A Machine Learning Perspective. M.S. thesis, University of Toronto. (2004)
- Maron M. and Kuhns J., On Relevance, Probabilistic Indexing and Information Retrieval. *Journal of the Association for Computing Machinery (JACM)*. vol 7, no 3, pages 216-244. (1960)
- Melville P. Mooney R. and Nagarajan R., Content-Boosted Collaborative Filtering for Improved Recommendations. In *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI'02)*. pages 187-192. Edmonton, Canada. (2001)
- Micarelli A., Gasparetti F., Sciarrone F. and Gauch S., Personalized Search on the World Wide Web. *The Adaptive Web*. pages 195-230. (2007)
- Miller B.N., Albert I., Lam S.K., Konstan J.A. and Riedl J., MovieLens Unplugged: Experiences with an Occasionally Connected Recommender System. In *Proceedings of the ACM 2003 International Conference on Intelligent User Interfaces (IUI'03)*. Pages 263-266. Miami, USA. (2003)
- Miller G.A., Beckwith R., Fellbaum C., Gross D. and Miller K., WordNet: An on-line Lexical Database. In *the International Journal of Lexicography*. vol 3, pages 235-244. (1990)
- Miller D., Leek T. and Schwartz R., A Hidden Markov Model Information Retrieval System. In *Proceedings of the 22nd ACM Conference on Research and Development in Information Retrieval (SIGIR'99)*. Pages 214-221. Berkeley, CA, USA. (1999)
- Mitra M., Singhal A., and Buckley C., Improving Automatic Query Expansion, In *Proceedings of the 21st Annual International ACM SIGIR Conference Research and Development in Information Retrieval*. pages 206-214. Melbourne, Australia. (1998)

- Mladenic, D. and Stefan J., Text-learning and Related Intelligent Agents: A Survey. *IEEE Intelligent Systems*. vol 14, no 4, pages 44-54. (1999)
- Mobasher B., Burke R., Bhaumik R. and Williams C., Toward Trustworthy Recommender Systems: An Analysis of Attack Models and Algorithm Robustness. *ACM Transaction on Internet Technology (TOIT)*. vol 7, no 4, Article No. 23. (2007)
- Moffat, A., and J. Zobel. Rank-biased Precision for Measurement of Retrieval Effectiveness. *ACM Transactions on Information Systems (TOIS)*. vol. 27, no 1, Article No 2. (2008)
- Murthi B.P.S. and Sarkar S., The Role of the Management Sciences in Research on Personalization. *In Journal of Management Science*. nol 49, no 10, pages 1344-1362. (2003)
- Mylonas P., Vallet D., Castells P., FernÁndez M. and Avrithis Y., Personalized Information Retrieval Based on Context and Ontological Knowledge. *Journal of The Knowledge Engineering Review*. vol 23, no 1, pages 73-100. (2008)
- Olivier B., The Unified Medical Language System (UMLS): Integrating Biomedical Terminology. *Nucleic Acids Research*. (2004)
- Pasquale L., Marco G. and Giovanni S., Content-Based Recommender Systems: State of the Art and Trends. *Recommender Systems Handbook*. pages 73-105. (2011)
- Paterek A., Improving Regularized Singular Value Decompositioin for Collaborative Filtering. In *Proceedings of the KDD Cup Workshop at the 13th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD'07)*. pages 39-42. San Jose, California, USA. (2007)
- Pazzani M., A Framework for Collaborative, Content-Based and Demographic Filtering. *Journal of Artificial Intelligence Review - Special Issue on Data Mining on the Internet*. Vol 13, No 5-6, pages 393-408. (1999)

- Pennock D., Horvitz E., Lawrence S. and Giles C.L., Collaborative Filtering by Personality Diagnosis: A Hybrid Memory- and Model-based Approach. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence (UAI' 2000)*. pages 473-480. Stanford, California, USA. (2000)
- Petros D., Iordanis K. and Prabhakar R., Competitive Recommender Systems. In *Proceedings of the 34th annual ACM symposium on Theory of Computing (STOC'02)*. pages 82-90. Montréal, Québec, Canada. (2002)
- Pickens J., Cooper M., Golovchinsky G., Reverted Indexing for Feedback and Expansion. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management (CIKM'10)*. pages 1049-1058. Toronto, Canada. (2010)
- Plachouras, V., He, B., & Ounis, I. Experiments in web, robust, and terabyte tracks with terrier. In *Proceedings of the 13th Text REtrieval Conference (TREC 2004)*. Gaithersburg, MD. (2004)
- Ponte J. and Croft W., A Language Modeling Approach to Information Retrieval. In *Proceedings of the 21st ACM Conference on Research and Development in Information Retrieval (SIGIR'98)*. pages 275-281. Melbourne, Australia. (1998)
- Popescul, A., Ungar, L.H., Pennock D.M and S. Lawrence. Probabilistic Models for Unified Collaborative and Content-Based Recommendation in Sparse-Data Environments. In *Proceedings of the 17th Uncertainty in Artificial Intelligence (UAI'01)*. pages 437-444. Seattle, Washington, UAS. (2001)
- Porter M.F., An algorithm for suffix stripping, In *Readings in Information Retrieval*, pages 313-316. (1997)
- Powell M.J.D., Approximation Theory and Methods. *Cambridge University*. Press. (1981)

- Qiu Y. and Frei H., Concept Based Query Expansion, In *Proceedings of the 16th International ACM SIGIR conference. Research and Development in Information Retrieval*, pages 160-169. Pittsburgh, PA, USA. (1993)
- Rabiner L., A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. In *A. Waibel and K. Lee (Eds.), Readings in Speech Recognition*. pages 267-296. (1990)
- Radev D.R., Jing H.Y., Styś M. and Tam D., Centroid-Based Summarization of Multiple Documents. In *Proceeding of NAACL-ANLP Workshop on Automatic Summarization*. vol 4, pages 21-30. Seattle, USA. (2000)
- Resnic P., Iacovou N, M Suchak M., Bergstrom P. and Riedl J., GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *Proceedings of the ACM Computer Supported Cooperative Work*. pages 175-186. New York, USA. (1994)
- Riedl M. and Biemann C., Text Segmentation with Topic Models. *Journal for Language Technology and Computational Linguistics*. vol 27, pages 47-69. (2012)
- Rich E. User Modeling Via Stereotypes. *Cognitive Science*. vol 3, no 4, pages 329-354. (1979)
- Robertson S.,E. The Probability Ranking Principle in Information Retrieval. *Readings in Information Retrieval*. pages 281-286. (1977)
- Robertson S.E. and Spärck Jones K., Relevance Weighting of Search Terms, *Document Retrieval Systems*. pages 143-160. (1976)
- Robertson S.E., Walker S., S. Jones, S., Hancock-Beaulieu, M.M. and Gatford, M., Okapi at TREC-3, In *Proceedings of the Second Text REtrieval Conference. (TREC-3)*. pages 109-126. (1995)

- Robertson S.E. and Walker S., Okapi, Keenbow at TREC-8. In *Proceedings of the 8th Text REtrieval Conference (TREC-8)*. pages 151-162. (2000)
- Rocchio J. and Salton G., Relevance Feedback in Information Retrieval. *The Smart Retrieval System-Experiments in Automatic Document Processing*. pages 313-323. (1971)
- Resnick P., Iacovou N., Suchak M., Bergstrom P. and Riedl J., Grouplens: An Open Architecture for Collaborative Filtering of Netnews. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW'94)*. pages 175-186. New York, USA. (1994)
- Sakai T., Robertson S.E., and Walker S., Flexible Pseudo-Relevance Feedback Via Direct Mapping and Categorization of Search Requests, In *Proceedings of the BCS-IRSG ECIR*, pages 3-14. Darmstadt, Germany. (2001)
- Salton G., The SMART Retrieval System—Experiments in Automatic Document Processing. *Book*. (1971)
- Salton G., Automatic Text Processing: the Transformation, Analysis, and Retrieval of Information by Computer. (1989)
- Salton G. and Buckley C., Improving Retrieval Performance by Relevance Feedback. *Readings in Information Retrieval*. pages 355-364. (1990)
- Salton G. and McGill M., Introduction to Modern Information Retrieval. (1983)
- Salton G. and Yang C., On the Specification of Term Values in Automatic Indexing. In *the Journal of Documentation*. vol 29, pages 351-372. (1973)
- Sandvig J.J., Mobasher B. and Burke R., Robustness of Collaborative Recommendation Based on Association Rule Mining. In *Proceedings of the ACM Conference on Recommender Systems (RecSys'07)*. pages 105-112. New York, USA. (2007)

- Sarwar B.M., Karypis G., Konstan J.A. and Riedl J., Item-Based Collaborative Filtering Recommender Algorithms. In *Proceedings of the 10th International Conference on World Wide Web (WWW'10)*. pages 285-295. (2010)
- Schein A.I., Popescul A., Ungar L.H. and Pennock D.M., Methods and Metrics for Cold-Start Recommendations. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2002)*. pages 253-260. Tampere, Finland. (2002)
- Scott M.L., Dewey Decimal Classification. 21st Edition. *Book*. (1998)
- Shardanand U. and Maes P., Social Information Filtering: Algorithms for automating “Word of Mouth”. In *Proceedings of the Conference Human Factors in Computing Systems the SIGCHI (CHI'95)*. pages 210-217. Denver, Colorado, USA (1995)
- Sheldon D., Shokouhi M., Szummer M. and Craswell N., LambdaMerge: Merging the Results of Query Reformulations. In *the Proceeding of the 4th ACM International Conference on Web Search and Data Mining (WSDM'11)*. Pages 795-804. HongKong. (2011)
- Shaw J.A. and Fox E.A., Combination of Multiple Searches. In *Proceedings of Second Text REtrieval Conference (TREC-2)*. page 243. (1994)
- Si L. and Jin R., A Flexible Mixture Model for Collaborative Filtering. In *the Proceeding of the 20th International Conference on Machine Learning (ICML'03)*. Pages 704-711. Washington DC, USA. (2003)
- Singha A., Modern Information Retrieval: A Brief Overview. *Bulletin of IEEE Computer Society Technical Committee on Data Engineering*. (2001)
- Smyth B., Briggs P., Coyle M. and O'Mahony M., Google Shared. A Case Study in Social Search. In *Proceedings of 17th International Conference on User Modeling Adaptation and Personalization: Formerly UM and AH (UMAP'09)*. pages 283-294. Trento, Italy.(2009)

- Soboroff and Nicholas C., Combining Content and Collaboration in Text Filtering. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI) Workshop: Machine Learning for Information Filtering*. Stockholm, Sweden. (1999)
- Soboroff I., Nicholas C., Collaborative Filtering and the Generalized Vector Space Model. In *Proceedings of the 23rd annual international ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2000)*. pages 351-353. Athens, Greece. (2000)
- Speretta M. and Gauch S., Personalized Search based on User Search Histories. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence (WI'05)*. pages 622-628. Compiegne University of Technology, France, (2005)
- Su X.Y. and Khoshgoftaar T.M., A Survey of Collaborative Filtering Techniques. *Journal of Advances in Artificial Intelligence*. Article No 4. (2009)
- Sugiyama K., Hatano K. and Yoshikawa M., Adaptive Web Search Based on User Profile Constructed without Any Effort from Users. In *Proceedings of the 13th International Conference on World Wide Web (WWW2004)*. pages 675-684. New York, USA, (2004)
- Tague J., Nelson, M., and Wu, H. Problems in the Simulation of Bibliographic Retrieval Systems. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1981)*. pages 236-255. Cambridge, UK. (1980)
- Tao T. and Zhai C., Regularized Estimation of Mixture Models for Robust Pseudo-Relevance Feedback. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. pages 162-169. Seattle, Washington, USA. (2006)
- Teevan J., Dumais S. and Horvitz E., Personalizing Search via Automated Analysis of Interests and Activities. In *Proceedings of the 28th Annual International ACM SIGIR*

- Conference on Research and Development in Information Retrieval (SIGIR 2005)*. pages 449-456. Salvador, Brazil. (2005)
- Ungar L. and Foster D., Clustering Methods for Collaborative Filtering. In *Proceedings of the Workshop on Recommender Systems*. AAAI Press, Menlo Park, CA. (1998)
 - Voorhees, E. M., Query expansion using lexical-semantic relations. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '94)*. pages 61-69. New York, NY, USA (1994)
 - Wan X.J. and Yang J.W., Multi-Document Summarization Using Cluster-Based Link Analysis. In *Proceedings of the Thirty-First Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'08)*. pages 299-306. Singapore. (2008)
 - Wang J., Language Models of Collaborative Filtering. In *Proceedings of the 5th Asia Information Retrieval Symposium on Information Retrieval Technology (AIRS 009)*. pages 218-229. Sapporo, Japan. (2009)
 - Wang J., Robertson S., Vries A.P., and Reinders M.J., Probabilistic Relevance Ranking for Collaborative Filtering. *Journal of Information Retrieval*. vol 11, no 6, pages 477-497. (2008)
 - Wen J.-R., Nie J.Y. and Zhang H.J., Query Clustering Using User Logs. In *Journal of ACM Transaction on Information Systems (TOIS)*. Vol. 20, No 1, Pages 59-81. (2002)
 - Xu J. and Croft W.B., Improving the Effectiveness of Information Retrieval with Local Context Analysis. *ACM Transaction on Information Systems (TOIS)*. vol. 18, no. 1, pages 79-112. (2000)
 - Xu J.X. and Croft W.B., Query Expansion Using Local and Global Document Analysis. In *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR 1996)*. pages 4-11. Zurich, Switzerland. (1996)

- Xue G.R., Lin C., Yang Q., Xi W., Zeng H.J., Yu Y. and Chen Z., Scalable Collaborative Filtering Using Cluster-Based Smoothing. In *Proceedings of 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (SIGIR'05). pages 114-121. Salvador, Bahia, Brazil. (2005)
- Xue G.R., Li H., Yang Q., Zeng H.J., Yu Y. and Chen Z., Exploiting the Hierarchical Structure for Link Analysis. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (SIGIR'05). Pages 186-193. Salvador, Bahia, Brazil. (2005)
- Xu Y., Jones G.J.F. and Wang B., Query Dependent Pseudo-Relevance Feedback Based on Wikipedia. In *Proceedings of 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. pages 59-66. Boston, USA. (2009)
- Yu J., Liu F.F. and Zhao H.H., Building User Profile based on Concept and Relation for Web Personalized Services. In *International Proceedings of Computer Science & Information Tech.* vol 36, page 172. (2012)
- Yurekli B., Capan G., Yilmazel B. and Yilmazel O. Guided Navigation Using Query Log Mining through Query Expansion. In *Proceedings of the 3rd International Conference on Network and System Security (NSS'09)*. pages 560-564. Gold Coast, Australia. (2009)
- Zhai C. and Lafferty J., A Study of Smoothing Methods for Language Models Applied to Information Retrieval. *ACM Transactions on Information System (TOIS)*. vol 22, no 2, pages 179-214. (2004)
- Zhang B., Li H., Liu Y., Ji L., Xi W., Fan W., Chen Z. and Ma W.Y., Improving Web Search Results Using Affinity Graph. In *Proceedings of the Twenty-Eighth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (SIGIR'05). pages 504-511. Salvador, Brazil. (2005)

- Zhou D., Orshanskiy A. Zha H. and Giles C.L.: Co-ranking Authors and Documents in a Heterogeneous Network. In *Proceedings of the 7th IEEE International Conference on Data Mining (ICDM'07)*. pages 739-744. Omaha, Nebraska, USA. (2007)

Appendix A: Topic Categories and Queries for the Extended FIRE 2011 Test Collection

Note: * mark indicates test queries which were randomly selected from each of the topic categories.

1. Indian armed forces	Weaponry Indian army Indian military * Indian air force Indian navy Women in Indian military Indian female soldiers
2. Bollywood movies	Latest movies * Ravan
3. Terrorist attacks	Terror attacks * Attacks in indian
4. Indian paintings and painters	Santiniketan Nandalal bose Chittoprasad Bikash Bhattacharyya Ancient Indian painters * Powder paint Powder paint creation ochre Technique powder paint Nature morte Painter court jury Indian painter exhibition Kangra paintings Hussain paintings MF Hussain controversy MF Hussain controversy paintings Posters for Bollywood movies
5. Indian education policy	Indian education policy Free school education in India Limit teacher vacancies * State government poser on varsity vacancies Varsity unrest over vacancy Education initialtives new delhi Child education Indian education

6. Indian tourism	Sightseeing and tourism in India Places to visit in India Touristic sights in India * Tourism Bengal Good places in India Tourist place adventure sport Tourist place adventure spot budget place Tourist place adventure spot budget place Haldia Taj mahal Historical places in India Top tourist attractions in India Tourist footfalls in India Must see tourist spots in India South India places Indian accommodation Indian folkways Mumbai tourism
7. Social impact on land acquisition	Ramesh nagar land acquisition delhi Land acquisition in India * People reaction to land acquisition in Bengal Farmers sad about land acquisition Bengal
8. Adventure sports	Adventure sport India Rock climbing in India * Bungee jumping
9. Honour killing incidents	Gaurav Honour killing cases * Bird flu Myanmar Killing unfaithful women honour Women honour killing case Origins of honour killings
10. Healthcare facilities	Healthcare Bengal Health care policy Dysentery Healthcare business Healthcare policy * Healthcare industry Dysentery flood rural region Dysentery flood rural region Balasore medical help Hospital facility Dental clinic Public hospital service
11. Relation of India with its neighbouring countries	India Pakistan relations bitter India Pakistan relations bitter kashmir issue trade India Pakistan relations bitter Kashmir issue terrorism Tibet new year celebration Tibet new year celebration indian government Sino Indian relations India China conflicts *

	India China border conflict Relation between indian and China The relation between india and Pakistan Relation between india and Burma The relation between india and Bangladesh India Pakistan incidents
12. Indian cricketing events	Indian cricket team performance Indian cricket team performance ganguly dhoni Indian cricket team better leader dhoni ganguly Englan vs Inda series World cup victory of India * Sachin Tendulkar Richie benaud Sachin Tendulkar Richie benaud world team Dravid wicket keeping ICC ganguly Dravid wicket keeping natwest final Memorable cricket events in India India's first test win India's first test win in 1950s Indian cricketing champion
13. Indian traditions and customs	Indian marriage customs Indian marriage events Indian new year * Indian new year celebration Indian wedding ceremony Indian wedding ceremony food Indian yoga Indian yoga breathing Traditional indian food Traditional indian wedding
14. History of Indian vernaculars	Indian vernaculars History of Indian languages * Tamil issue of classical language Telegu and kannada as classical languages
15. Indian political scams	Indian slum * Indian political scams Indian political scam cases
16. Indian Religion	The number of indian religions * Hindi in India Religion conflicts in India
17. Indian Election	Indian president Indian election scandal *
18. Racism	Racism in india Conflicts for racism in India * People reaction to racism in India
19. Scandals in Indian	Scandal in Indian Political scandal in india * Famous people scandals india
20. Border Trade of Indian with	Border trade of india with China

Its Neighbours	Border trade of india with Pakistan * Silk route of border trade Importance of border trade in india The good for border trade
21. Space Exploration	Indian space exploration * Indian space techniques
22. Disasters in Indian	Flood in india Typhoon in india Tsunami in india *
23. Indian Hockey Event	Indian hockey stars * Indian hockey champion
24. International Economic Slump	Effect of international economic slump in indan Food price in indian during internation economic slump Gold price in recession *
25. Indian culture	Indian culture * Indian religion cultures
26. Indian women	Indian women human rights Indian women dress * Indian women gold accessory Indian women sari
27. Indian population problem	Indian population * Indian child education problem Effect of indian population problem

Appendix B: Instructions for Participants in the FIRE PIR Task

For this search exploration task, you can search in a collection of Indian newspaper articles (Telegraph and BDNews 24 - Document News Collection 2001-2010).

- You can select a search category by clicking on “Select Category”.
- To search for documents, enter your query in the input field and click “Search”.
- You can view documents by clicking on the links in the result list.
- You can bookmark a document by clicking on the star next to its name.
- You can review previous results by clicking on “View Results”.
- To finish the exploratory search and submit, you must click “Finish Browsing” and then fill in the provided Topic Submission Form.
- For every topic you need to provide a title (typically 2 or 3 words closely resembling a web search query), a description (typically one sentence) and a narrative (2 or 3 sentences describing characteristics that make a document relevant or irrelevant). You can also summarize your view on the selected topic. For example, if you want to find information on "Osteoporosis", your title can be "Osteoporosis", the description can be "Osteoporosis bone disease" and the narrative might be "Documents discussing the disease are relevant. Documents about medicines are relevant. Documents on other diseases are not relevant here."
- You may then proceed to choose further categories.

Appendix C: Publications

- Mulwa C., Li W., Lawless S. and Jones G.J.F., A Proposal for the Evaluation of Adaptive Information Retrieval Systems using Simulated Interaction. In Proceedings of the Workshop on Simulation of Interaction: Automated Evaluation of Interactive IR at SIGIR 2010, Geneva, Switzerland, July 2010.
- Li W., Min J. and Jones G.J.F., A Text-Based Approach to the ImageCLEF 2010 Photo Annotation Task. CLEF (Notebook Papers/LABs/Workshops). (2010)
- Ganguly D., Leveling J., Li W. and Jones G.J.F., Towards Evaluation of Personalized and Collaborative Information Retrieval. In Proceedings of the First Workshop on Personalised Multilingual Hypertext Retrieval (PMHR 2011), Eindhoven, The Netherlands, June 2011.
- Li W., Ganguly D. and Jones G.J.F., Enhanced information retrieval using domain-specific recommender models . In Proceedings of the 3rd International Conference on the Theory of Information Retrieval (ICTIR'11), Bertinoro, Italy, September 2011.
- Ganguly D., Leveling J., Curtis K., Li W. and Jones G.J.F., Overview of the Personalised and Collaborative Information Retrieval (PIR) Track at FIRE 2011 , In Proceedings of the Third Workshop of the Forum for Information Retrieval (FIRE 2011), Mumbai, India, December 2011.
- Li W. and Jones G.J.F, Utilizing Recommender Algorithms for Enhanced Information Retrieval, In Proceedings of OAIR 2013 - 10th International Conference in the RIAO series, Lisbon, Portugal, May 2013.

- Li W. and Jones G.J.F., Enhanced Information Retrieval by Exploiting Recommender Techniques in Cluster-Based Link Analysis, In Proceedings of the Fourth International Conference on the Theory of Information Retrieval (ICTIR 2013), Copenhagen, Denmark, September 2013.
- Zhou D., Truran M., Liu J.X., Li W. and Jones G.J.F., Iterative Refinement Methods for Enhanced Information Retrieval. In Journal of International Journal of Intelligent Systems. Vol 29, No 4. Page 341-364. April 2014.