Mining Specific and General Features in Both Positive and Negative Relevance Feedback

QUT E-Discovery Lab at the TREC'09 Relevance Feedback Track

Yuefeng Li, Xiaohui Tao, Abdulmohsen Algarni[†],
Sheng-Tang Wu[*]
School of Information Technology, Queensland University of Technology, Australia
{y2.li, x.tao}@qut.edu.au, [†]abdulmohsen.algarni@student.qut.edu.au
[*]Dept. of Information Science and Applications, Asia University, Taiwan, swu@asia.edu.tw

Abstract

User relevance feedback is usually utilized by Web systems to interpret user information needs and retrieve effective results for users. However, how to discover useful knowledge in user relevance feedback and how to wisely use the discovery knowledge are two critical problems. In TREC 2009, we participated in the Relevance Feedback Track and experimented a model consisting of two innovative stages: one for subject-based query expansion to extract pseudo-relevance feedback; one for relevance feature discovery to find useful patterns and terms in relevance judgements to rank documents. In this paper, the detailed description of our model is given, as well as the related discussions for the experimental results.

1 Introduction

Web users' personal interests and preferences can be drawn in their user profiles. In Web information gathering, user profiles are used by many works to search information for users according to their personal needs [3, 10]. However, effectively acquiring user profiles is difficult. To acquire user profiles, some techniques explicitly interview users [13], some use user relevance feedback [14]. These mechanisms require user-effort in the user profile acquisition process. Attempting to release such burden from users, alternatively some automatic techniques have been developed to acquire user profiles from a collection of user personal information, for example, browsing history [3, 17]. User profiles acquired by such techniques, however, usually contain noise and uncertainties. Hence, a method to acquire user profiles effectively and efficiently (without the burden of user-effort) is an urgent need for personalized Web information gathering.

Relevance features describe what a user wants. They can be discovered from user relevance feedback. Over the years, pattern-based approaches have been expected to outperform term-based techniques when discovering relevance features. Patterns are more discriminative and carry more "semantics". However, according to information retrieval (IR) experiments, few significant improvements have been made by using pattern-based methods to replace term-based methods [15,16]. When utilizing pattern mining techniques, people encountered two problems: (i) high frequent patterns are usually general, whereas specific patterns are usually with low frequency (this is because the measuring methods for pattern learning, such as "support" and "confidences", appeared unsuitable in the filtering stage [11]); (ii) negative user feedback is difficult to use when revising the features extracted from the positive user feedback. Relevance feature discovery is challenging [10, 12].

Motivated by these challenges, we proposed a relevance feature discovery model and tested the model in the Relevance Feedback track in TREC 2009. This Relevance Feedback track was designed to evaluate a system's capacity of finding quality user relevance feedback, as well as its relevance feedback algorithms. Thus, two phases were conducted in the track corresponding to this design: (i) identifying a small number of documents for (pseudo) relevance feedback; (ii) running relevance feedback algorithms with relevance judgements. In accordance to the two phases, we participated with also a two-stage information filtering model: (i) subject-based query expansion for pseudo relevance feedback extraction; (ii) pattern-based relevance feature discovery using both positive and negative feedback. The model aimed to discover relevance features for Web user profile acquisition.

The first stage was to expend a query (topic) to retrieve pseudo relevance feedback. To expand queries, we used a subject ontology LCSH (Library of Congress Subject Head-

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
NOV 2009		**00-00-2009 to 00-00-2009**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Mining Specific and General Features in Both Positive and Negative Relevance Feedback. QUT E-Discovery Lab at the TREC'09 Relevance Feedback Track	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Queensland University of Technology,School of Information Technology,Australia,**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
Proceedings of the Eighteenth Text REtrieval Conference (TREC 2009) held in Gaithersburg, Maryland, November 17-20, 2009. The conference was co-sponsored by the National Institute of Standards and Technology (NIST) the Defense Advanced Research Projects Agency (DARPA) and the Advanced Research and Development Activity (ARDA).

14. ABSTRACT
see report

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**12**	

ings). The ontology specified commonsense knowledge obtained by people through their experience and education, and was successfully evaluated in our prior work reported in [18]. Given a query, the topic-related subjects were extracted from the LCSH ontology. On the basis of these subjects, user background knowledge was discovered and a personalized ontology was constructed. Based on the personalized ontology and using an information gathering system, a training set (consisting of a positive and a negative subsets) was extracted from the ClueWeb09 Category-B corpus based on title search, and treated as pseudo relevance feedback.

At the second stage, relevance features were discovered from both positive and negative pseudo relevance feedback, using a model introduced in [9]. These relevance features consisted of high-level pattern features and low-level term features. Based on the high-level features, the low-level features were classified into three groups: positive specific terms, general terms, and negative specific terms. When applying negative patterns to revise the discovered features, we increased the weight of positive specific terms but declined that of negative specific terms. This feature revision went into a loop to optimize the relevance feature extraction. Finally, documents highly relevant to these relevance features were retrieved from the ClueWeb09 Category-B as the final submission results.

In this paper, the two-stage model and the related evaluation in TREC 2009 Relevance Feedback track are presented and discussed. Section2 introduces the subject-based query expansion, and Section 3 presents relevance feature discovery using positive and negative samples. After that, the evaluation results are discussed in Section 4. Finally, the last section makes conclusions.

2 Subject-Based Query Expansion for Pseudo Relevance Feedback

The first stage aims to automatically retrieve pseudo relevance feedback from the ClubWeb09 Category-B. Because there was only a limited number of terms in given topics, the key issue here was how to acquire user interest from the limited information. In this work, we utilized a world knowledge ontology to analyze the concepts in the given topics. For an incoming topic, the positive subjects were extracted from the ontology. Based on these subjects and their referring-to instances, user background knowledge was discovered and utilized to expand the given query terms and to search the ClueWeb09 Category-B for pseudo relevance feedback. The top five ranked results were considered relevance feedback from users. Figure 1 illustrates the architecture of our Stage 1 process.

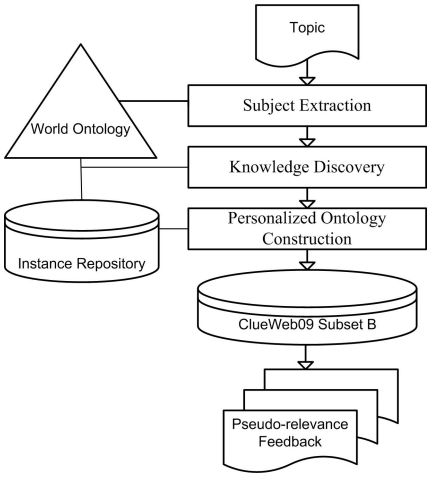


Figure 1. The Stage 1 Architecture

2.1 World Ontology and Instances

The world ontology was encoded from the Library of Congress Subject Headings[1], a library catalog system. The LCSH system is a categorization developed for organizing the large volumes of library collections and for retrieving information from the library. The references specified in LCSH for subject headings were encoded into the semantic relations associated with and linking the subjects, where *Broader term/Narrower term* were for *is-a*, *Used-for* for *part-of*, and *related-to* for *related-to* relations. The LCSH ontology contained about 400,000 topical, geographical, and corporate subjects.

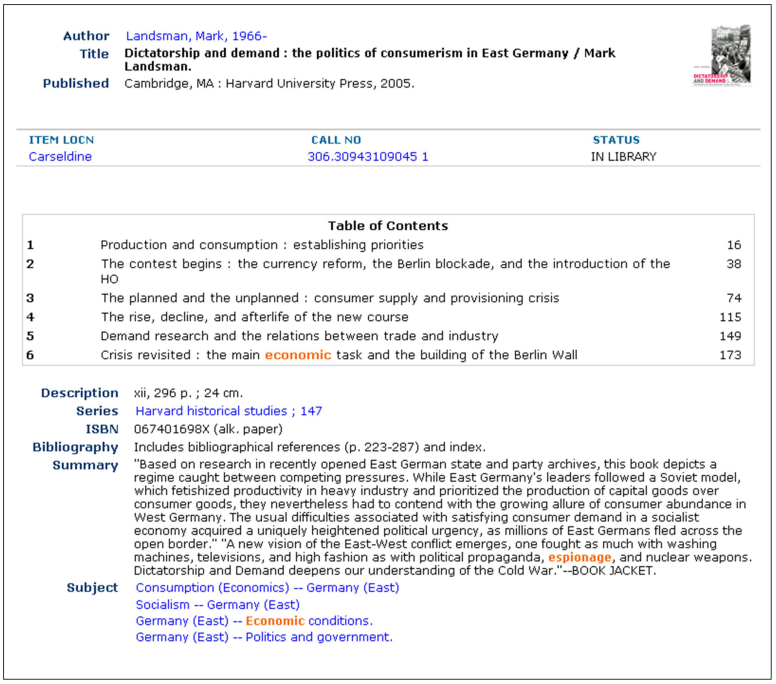
The LCSH ontology was populated using the instances encoded from the information items in a library catalog [2]. Figure 2 illustrates a sample information item for instances. The descriptive information, such as the title and table of contents, are the knowledge resource extensive from the LCSH ontology. Such descriptive information was used for the content of an instance. A list of indexed content-based descriptors (subjects) is cited by each item (instance). Thus, we could have a matrix constructed by instances and subjects. Each instance may cite a list of subjects, and each subject may refer to a list of instances. Based on this matrix, the belief (bel) of an instance to a subject can be determined:

$$bel(i,s) = \frac{1}{index(s,i) \times |\eta(i)|};$$

where $\eta(i)$ is the set of subjects cited by i, $index(s,i)$ is the index (starting with one) of s on the citing list. Us-

[1] http://classificationweb.net/.

[2] In particular, the QUT library. For the sake of simplicity, only the abstracted information (title, table of content, and summary) was used to represent an instance. Example of instances can be found on http://www.library.qut.edu.au.



Author Landsman, Mark, 1966-
Title Dictatorship and demand : the politics of consumerism in East Germany / Mark Landsman.
Published Cambridge, MA : Harvard University Press, 2005.

ITEM LOCN	CALL NO	STATUS
Carseldine	306.30943109045 1	IN LIBRARY

Table of Contents

1	Production and consumption : establishing priorities	16
2	The contest begins : the currency reform, the Berlin blockade, and the introduction of the HO	38
3	The planned and the unplanned : consumer supply and provisioning crisis	74
4	The rise, decline, and afterlife of the new course	115
5	Demand research and the relations between trade and industry	149
6	Crisis revisited : the main economic task and the building of the Berlin Wall	173

Description xii, 296 p. ; 24 cm.
Series Harvard historical studies ; 147
ISBN 067401698X (alk. paper)
Bibliography Includes bibliographical references (p. 223-287) and index.
Summary "Based on research in recently opened East German state and party archives, this book depicts a regime caught between competing pressures. While East Germany's leaders followed a Soviet model, which fetishized productivity in heavy industry and prioritized the production of capital goods over consumer goods, they nevertheless had to contend with the growing allure of consumer abundance in West Germany. The usual difficulties associated with satisfying consumer demand in a socialist economy acquired a uniquely heightened political urgency, as millions of East Germans fled across the open border." "A new vision of the East-West conflict emerges, one fought as much with washing machines, televisions, and high fashion as with political propaganda, espionage, and nuclear weapons. Dictatorship and Demand deepens our understanding of the Cold War."--BOOK JACKET.
Subject Consumption (Economics) -- Germany (East)
Socialism -- Germany (East)
Germany (East) -- Economic conditions.
Germany (East) -- Politics and government.

Figure 2. An Instance from A Library Catalog Item

ing the instance displayed in Fig. 2 as a sample, let i be this instance; s be the subject *Consumption (Economics)–Germany (East)*. We have $index(s, i) = 1$ and $|\eta(i)| = 4$, and can thus calculate $bel(i, s) = 0.25$. The less subjects cited by an instance and the higher index a subject on a citing list, the stronger belief the instance holds to the subject. The $bel(i, s)$ will be used to select the right instances to populate the LCSH ontology.

A method, specificity [18, 19] (denoted as *spe*), was further utilized to measure the focus of a subject in the LCSH ontology. The subjects located at upper bound levels in the ontology are more abstractive than those at lower bound levels towards the "leaves". Also, upper bound level subjects have more descendant subjects in shadow, in comparison with lower bound level subjects. Thus, an upper bound subject has weaker focus than a lower bound subject in its shadow.

The spe value of a subject s is determined by analyzing its associated hierarchical relations of *is-a* and *part-of*. By setting the spe value for "leave" subjects as 1, toward the root of the ontology, the spe value decreases for each level up. If a subject has all direct child subjects in shadow with *is-a* relationship, the smallest spe of its child subjects is chosen for the subject's spe value by decreasing 10%. If a subject has all direct child subjects in shadow with *part-of* relationship, its spe is defined as the average spe value of its child subjects, applying the 10% decreasing rate. If the direct child subjects in shadow are mixed with *is-a* and *part-of* relations to their parent subject, two spes are calculated: one for *is-a* child subjects, and one for *part-of* subjects. The smaller spe is then chosen to value the spe of the parent subject. As a result, the specificity of a upper bound subject is guaranteed smaller than that of a lower bound subject in its shadow.

2.2 Interesting Subject Discovery

Given a topic $\mathcal{T} := \{t_1, t_2, \ldots, t_n\}$, two sets of subjects were extracted from the LCSH ontology: positive subjects $\mathcal{S}^+$ being relevant to the topic; and negative subjects $\mathcal{S}^-$ being paradoxical or ambiguous to the topic. If a subject's label contains any keywords in the topic ($label(s) \cap \mathcal{T} \neq \emptyset$), this subject is extracted and put into the initial positive subject set ($\mathcal{S}^+ = \mathcal{S}^+ \cup \{s\}$). The positive level of s to $\mathcal{T}$ is thus measured by

$$pos(s, \mathcal{T}) = spe(s) \times |label(s) \cap \mathcal{T}| \times \sum_{i \in \eta^{-1}(s)} sup(i, \mathcal{T})$$

where

$$sup(i, \mathcal{T}) = \sum_{s' \in \eta(i)} bel(i, s') \times |label(s') \cap \mathcal{T}|$$

as defined previously, $\eta(i)$ refers to the set of subjects cited by i, and $\eta^{-1}(s)$ gives the set of instances citing s.

The reachable ancestor and descendant subjects of s in the ontology were also extracted. The "reachable" here is

limited to the distance of three edges in the ontology. The subjects located more than that distance are unlikely important to $\mathcal{T}$, as reported by [6]. These reachable subjects were extracted and put into the negative subject set ($\mathcal{S}^-$).

User background knowledge was discovered from the reference between the subjects and their instances. Let $s_1 \in \mathcal{S}^+$ and $s_2 \in \mathcal{S}^-$. If $\eta^{-1}(s_1) \cap \eta^{-1}(s_2) \neq \emptyset$, s_1 and s_2 have something in common and are relevant. The certainty level of s_2 being positive was thus determined by its linked positive subjects (e.g. $s_1 \in \mathcal{S}^+$). A subject is more interesting if it has more linked positive subjects. Let $\widehat{S}(s)$ be the set of linked positive subjects of $s \in \mathcal{S}^-$, we measure the certainty level of s to $\mathcal{T}$ by:

$$pos(s, \mathcal{T}|s \in \mathcal{S}^-) = \frac{\sum_{s' \in \widehat{S}(s)} conf(s', s) \times pos(s', \mathcal{T})}{|\widehat{S}(s)|}$$

where

$$conf(s', s) = \frac{|\eta^{-1}(s') \cap \eta^{-1}(s)|}{\eta^{-1}(s')}$$

Considering such discovered user background knowledge, if a $s \in \mathcal{S}^-$ has $pos(s, \mathcal{T}) > 0$, it would be removed from $\mathcal{S}^-$ and replaced to $\mathcal{S}^+$.

2.3 Query Expansion for Pseudo Relevance Feedback Extraction

The query terms were expanded based on the positive subjects discovered in the previous section. In Section 2.2, a set of positive subjects $\mathcal{S}^+$ was discovered, in which each subject was assigned a pos value indicating the certainty level of the subject being relevant to the given topic. In Section 2.1, we know that a subject refers to a set of instances. Thus, a training set D^+ could be generated, in which each document d was from the content of an instance i referred to by a positive subject $s \in \mathcal{S}^+$. A $support$ value was calculated for each document in the training set, by accumulating all pos values of the subjects on the citing list of the instance. The expanding terms were extracted from the training set.

The training set was first used to evaluate weights for a set of selected terms T. After text pre-processing of stopword removal and word stemming, the semantic space referred to by a d was represented by its normal form $\beta(d) = \{(t_1, w_1), (t_2, w_2), \ldots, (t_k, w_k)\}$, where w is the weight distribution of terms and $w_i = \frac{f_i}{\sum_{j=1}^{k} f_j}$ and f_i is the term frequency of t_i in d. A probability function on T was derived based on the normal form of positive documents and their supports for all $t \in T$:

$$pr_\beta(t) = \sum_{d \in D^+, (t,w) \in \beta(d)} support(d) \times w$$

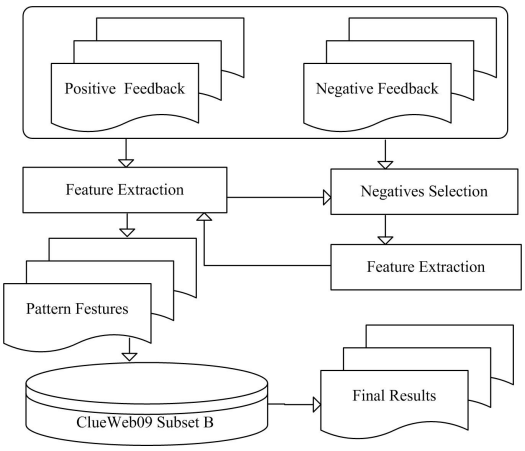


Figure 3. The Stage 2 Architecture

The terms with top 150 $pr_\beta(t)$ values were then selected to expand the query terms given in $\mathcal{T}$. The details of evaluation can be referred to [10].

The documents in the ClueWeb09 corpus were indexed by accumulating the $pr_\beta(t)$ of the expanded top 150 terms that occurred in the document titles. Because ClueWeb09 Category-B is a large corpus, in order to reduce the complexity, only the title of documents counted into this index calculation. The top five indexed documents were chosen as the pseudo relevance feedback from users, and submitted as the results for Phase 1 of the track.

3 Relevance Feature Discovery

Relevance feature discovery aims to discover a set of features from text documents to describe what a user wants. In Phase 2 of TREC'09 Relevance Feedback track, a given topic was represented by a set of user judgements containing documents associated with values of 0, 1, or 2, indicating being non-relevant, relevant, and highly relevant to the topic, respectively. Treating the documents associated with 1 and 2 as equally positive and those with 0 negative, we had two different sets: positive and negative feedback. In this Stage 2 method, relevance features were to be discovered from both of the positive and negative relevance feedback.

When generating the positive and negative feedback, two special problems were encountered: (i) positive feedback was unavailable because all judgements were with 0 (non-relevant). For this problem, we formed a positive document by using the query terms expanded in Stage 1 (as discussed in Section 2.3), and weighted these terms equally as 1; (ii) negative feedback was unavailable because all judgement were with 1 or 2. For this problem, we used only positive feedback for feature discovery.

Table 1. A set of paragraphs

Paragraph	*Terms*
dp_1	$t_1\ t_2$
dp_2	$t_3\ t_4\ t_6$
dp_3	$t_3\ t_4\ t_5\ t_6$
dp_4	$t_3\ t_4\ t_5\ t_6$
dp_5	$t_1\ t_2\ t_6\ t_7$
dp_6	$t_1\ t_2\ t_6\ t_7$

The pattern-based features were first extracted from the positive user feedback. After that, these features were used to iteratively select and re-select meaningful negative documents (called offenders in this paper) from the negative feedback. These offenders were used to revise the extracted features. Finally, the revised features were used to retrieve the final results from the ClueWeb09 Subset-B. Figure 3 illustrates the architecture of our model in Stage 2.

3.1 Frequent and Closed Sequential Patterns

For a given topic, relevance feature discovery extracts from a document set a set of features, including patterns and terms, and assigns them weights. The document set, usually called a training set and denoted as D, consists of a set of positive documents (D^+) and a set of negative documents (D^-). When splitting a document into paragraphs, a document d can also be represented by a set of paragraphs $PS(d)$.

Let $T = \{t_1, t_2, \ldots, t_m\}$ be a set of terms extracted from D^+; X be a set of terms (called a *termset*) in document d. $coverset(X)$ denotes the covering set of X for d, which includes all paragraphs $dp \in PS(d)$ where $X \subseteq dp$, i.e., $coverset(X) = \{dp | dp \in PS(d), X \subseteq dp\}$. The *absolute support* of X is the number of occurrences of X in $PS(d)$: $sup_a(X) = |coverset(X)|$. The *relative support* of X is the fraction of the paragraphs that contain the pattern: $sup_r(X) = \frac{|coverset(X)|}{|PS(d)|}$. A termset X is then called a *frequent pattern* if its sup_a (or sup_r) $\geq min\_sup$, a minimum support.

Table 1 lists a set of paragraphs for a document d, where $PS(d) = \{dp_1, dp_2, \ldots, dp_6\}$ with duplicate terms removed. Assume $min\_sup = 3$, ten frequent patterns would be extracted as shown in Table 2.

Given a set of paragraphs $Y \subseteq PS(d)$, we can also define its *termset*, which satisfies

$$termset(Y) = \{t | \forall dp \in Y \Rightarrow t \in dp\}.$$

By defining the closure of X as:

$$Cls(X) = termset(coverset(X))$$

Table 2. Frequent patterns and covering sets

Frequent Pattern	*Covering Set*
$\{\mathbf{t_3}, \mathbf{t_4}, \mathbf{t_6}\}$	$\{dp_2, dp_3, dp_4\}$
$\{t_3, t_4\}$	$\{dp_2, dp_3, dp_4\}$
$\{t_3, t_6\}$	$\{dp_2, dp_3, dp_4\}$
$\{t_4, t_6\}$	$\{dp_2, dp_3, dp_4\}$
$\{t_3\}$	$\{dp_2, dp_3, dp_4\}$
$\{t_4\}$	$\{dp_2, dp_3, dp_4\}$
$\{\mathbf{t_1}, \mathbf{t_2}\}$	$\{dp_1, dp_5, dp_6\}$
$\{t_1\}$	$\{dp_1, dp_5, dp_6\}$
$\{t_2\}$	$\{dp_1, dp_5, dp_6\}$
$\{\mathbf{t_6}\}$	$\{dp_2, dp_3, dp_4, dp_5, dp_6\}$

a pattern (or termset) X is *closed* if and only if $X = Cls(X)$.

Let X be a closed pattern. We have

$$sup_a(X_1) < sup_a(X) \tag{1}$$

for all patterns $X_1 \supset X$.

A taxonomy can be constructed by using closed patterns with *is-a* (or *subset*) relations. Table 2 contains three closed patterns, $< t_3, t_4, t_6 >$, $< t_1, t_2 >$, and $< t_6 >$, within ten frequent patterns. After pruning the non-closed patterns, a pattern taxonomy PT can be constructed, like $PT = \{\langle t_3, t_4, t_6\rangle, \langle t_1, t_2\rangle, \langle t_6\rangle\}$ in Table 2 when considering $\langle t_6\rangle$ a subset of $\langle t_3, t_4, t_6\rangle$.

Small patterns (e.g. $\langle t_6\rangle$) in a taxonomy are usually general because they have more chance to be used frequently. Vice versa, large patterns (e.g. $\langle t_3, t_4, t_6\rangle$) are relatively specific because they usually have a low frequency.

A sequential pattern $s =< t_1, \ldots, t_r > (t_i \in T)$ is an ordered list of terms. Denoted by $s_1 \sqsubseteq s_2$, a sequence $s_1 =< x_1, \ldots, x_i >$ is a sub-sequence of $s_2 =< y_1, \ldots, y_j >$, iff $\exists j_1, \ldots, j_i$ such that $1 \leq j_1 < j_2 \ldots < j_i \leq j$ and $x_1 = y_{j_1}, x_2 = y_{j_2}, \ldots, x_i = y_{j_i}$. Given $s_1 \sqsubseteq s_2$, we call s_1 a sub-pattern of s_2, and s_2 a super-pattern of s_1. To simplify the explanation, we refer to sequential patterns as patterns.

As the same as those defined for normal patterns, we define the *absolute support* and *relative support* for a pattern (an ordered *termset*) X in d. We also denote the covering set of X as $coverset(X)$, which includes all paragraphs $ps \in PS(d)$ such that $X \sqsubseteq ps$, i.e., $coverset(X) = \{ps | ps \in PS(d), X \sqsubseteq ps\}$. X is then called a *frequent pattern* if $sup_r(X) \geq min\_sup$. By using Eq. (1), a frequent sequential pattern X is *closed* if $\nexists$ any super-pattern X_1 of X such that $sup_a(X_1) = sup_a(X)$.

3.2 Deploying High-Level Patterns on Low-Level Terms

To overcome the problem of patterns with low-frequency, a method was developed to deploy high level patterns over low-level terms. The evaluation of term supports (weights) in this paper is different from that in term-based approaches. For a term-based approach, the value of a term is scaled based on its appearance in documents. In our method, the value of terms are scaled based on their appearance in discovered patterns.

To improve the efficiency of the pattern taxonomy mining (PTM), an algorithm, *SPMining*(D^+, min_sup), was introduced by [21] and further developed in [11, 20] to find closed sequential patterns from positive documents D^+. The *SPMining* algorithm used the well-known *Apriori* property to narrow down the searching space.

Let SP_1, SP_2, ..., SP_n be the sets of discovered closed sequential patterns for all documents $d_i \in D^+ (i = 1, \cdots, n)$, where $n = |D^+|$. For a given term t, its *weight* in discovered patterns is assigned by:

$$w(t, D^+) = \sum_{i=1}^{n} \sum_{t \in p \subseteq SP_i} \frac{sup_r(p, d_i)}{|p|} \qquad (2)$$

where $|p|$ is the number of terms in p.

With weights assigned to the terms in D^+, a function can be used to rank and judge the relevance of incoming documents:

$$rank(d) = \sum_{t \in T} w(t)\tau(t, d)$$

where $w(t) = w(t, D^+)$; and $\tau(t, d) = 1$ if $t \in d$, otherwise $\tau(t, d) = 0$.

3.3 Mining Negative Patterns for Revising Low-Level Features

In general speaking, the definition of relevance is subjective. People may describe the relevance of a topic (or a document) in two dimensions, specificity and exhaustivity, where specificity describes the focus of the topic on what users want, and exhaustivity describes the extent of the topic dealing what users want. Such two-dimension description is easy for human beings to use, however, difficult for a computational system to apply. In this section, we first discuss how to use the two dimensions to understanding the semantic meanings of low-level feature terms. We also present an algorithm for negative pattern discovery and term weight revision.

3.3.1 Specific and General Features

Let DP^+ be the union of all patterns in pattern taxonomies discovered from D^+, and DP^- be the union of all negative patterns in the pattern taxonomies discovered from D^-. A closed sequential pattern of D^+ (or D^-) is called a *positive pattern* (or *negative pattern*).

Given a term $t \in T$, its *exhaustivity* refers to the number of discovered patterns containing t in both DP^+ and DP^-, and its *specificity* refers to the number of discovered patterns containing t in only DP^+ but not DP^-. Based on these, we can classify terms into three groups: *general terms* (GT,) for those appearing in both positive patterns and negative patterns; *positive specific terms* (T^+) for those appearing in only positive patterns; *negative specific terms* (T^-) for those appearing in only negative patterns. They are defined by:

$$GT = \{t | (\exists p_1 \in DP^+) \wedge (\exists (p_2 \in DP^-) \Rightarrow t \in (p_1 \cap p_2)\},$$

$$T^+ = \{t | t \notin GT, \exists (p \in DP^+) \Rightarrow t \in p\}, \; and$$

$$T^- = \{t | t \notin GT, \exists (p \in DP^-) \Rightarrow t \in p\}$$

where $GT \cap T^+ \cap T^- = \emptyset$.

Specific terms contain more semantic meanings and distinguish a topic from others. Thus, specific terms are useful to describe the relevance feature of a topic. However, using specific terms alone may be insufficient when trying to improve the performance of relevance feature discovery. Documents containing no specific terms may also highlight user information needs as well. Therefore, one possible solution is to use the hybrid of specific terms, general terms, and negative terms. However, adequate control is necessary for the side effects generated by using general terms.

3.3.2 Revision Strategy

In this section, we discuss the basic strategies of revising the features discovered from a training set. This feature revising process takes place only after terms are classified into three categories of *general*, *positive specific*, and *negative specific terms*.

From the positive documents in a training set, the revising process first discovers initial positive features including high-level positive patterns and low-level terms. Selecting some negative samples from the negative documents in the training set, the process also discovers negative patterns and terms by using the same pattern mining technique as that used for positive feature discovery. The process then revises the initial features to obtain revised features. This process can be repeated several times: selecting negative documents, mining negative features and revising revised features.

Algorithm *NFMining*(D) describes the details of the the revision strategy, with an assumption that the number of negative documents is greater than the number of positive documents. For a given training set $D = \{D^+, D^-\}$, we assume that the initial features, (DP^+, DP^-, T), have been

Algorithm 1. ***NFMining***(D)

Input: A training set, $\{D^+, D^-\}$, $\alpha = -1$;
extracted features (DP^+, DP^-, T), $DP^- = \emptyset$;
support function, minimum support min_sup,
and experimental parameters K and σ.
Output: Updated term set T and function *weight*.

Method:
1: $GT = \emptyset, T^+ = \emptyset, T^- = \emptyset, loop = 0$;
2: **foreach** $t \in T$ **do**
3: $weight(t) = weight(t, D^+)$;
4: **foreach** $d \in D^-$ **do**
5: $rank(d) = \Sigma_{t \in d \cap (T \cup T^-)} weight(t)$;
6: let $D^- = \{d_0, d_1, ..., d_{|D^-|-1}\}$ in descendent order,
let $j = \sigma$ if $loop = 0$, otherwise $j = 0$;
7: $D_3^- = \{d_i | d_i \in D^-, j \leq i < K + j\}$;
8: DP^- =*SPMining*(D_3^-, min_sup); //find negative patterns
9: $T_0 = \{t \in p | p \in DP^-\}$; // all terms in negative patterns
10: **foreach** $t \in (T_0 - T)$ **do**
11: **if** $(loop = 0)$ **then** $weight(t) = \alpha \times weight(t, D_3^-)$
else $weight(t) = \alpha \times weight(t, D_3^-) + weight(t)$;
12: $T^- = T^- \cup (T_0 - T)$, $loop++$;
13: **if** $loop < 3$ **then** goto step 4;
14: **foreach** $t \in T$ **do** //term partition
15: **if** $(t \in T^-)$ **then** $GT = GT \cup \{t\}$
else $T^+ = T^+ \cup \{t\}$;
16: **foreach** $t \in T^+$ **do**
17: $weight(t) = weight(t) \times (1 + \frac{|\{d | d \in D^+, t \in d\}|}{|D^+|})$;
18: $T = T \cup T^-$;

Table 3. Example of a set of terms discovered from DP^+, $DP^+ \in D^+$ and $|D^+| = 6$.

term	*weight*	*# of docs that include the term*
$\langle t_1 \rangle$	0.34	4
$\langle t_2 \rangle$	0.90	6
$\langle t_3 \rangle$	0.55	3
$\langle t_4 \rangle$	0.65	5
$\langle t_5 \rangle$	0.75	6
$\langle t_6 \rangle$	0.84	2

extracted from positive documents D^+ before the algorithm starts, where $T = \{t \in p | p \in DP^+\}$ and $DP^- = \emptyset$. The experimental parameter is set as $\alpha = -1$ to calculate the weights of terms in negative patterns.

Step 1 initializes the sets of general terms GT, positive specific terms T^+, and negative specific terms T^-. $loop$ is used to control the number of revision cycles. Step 2 and 3 compute weights for all terms in T. Table 3 shows a set of terms and their weights deploying from positive patterns. In experiments, when positive documents were unavailable, a set of 100 terms with weight set to 1 from query expansion (as discussed in Section 2.3) were used as positive terms.

Steps 4 and 5 rank documents in the negative document set. If t is a negative specific term, its has an revising weight evaluated in step 10 and 11. The weight function is described as:

$$weight(t) = \begin{cases} \text{its revising weight}, & \text{if } t \in T^- \\ support(t, D^+), & \text{otherwise} \end{cases}$$

Steps 6 and 7 sort the negative documents based on their rank values, and select offenders (meaningful negative documents). A document is considered negative to the topic if it is ranked lower than or equal to 0. For the first loop the minimum weight that we can get is 0 because there is no negative weight in the term set T. However, from the next loop some negative terms from D^- with negative weight are added. Then it is most likely to get weight less than 0. If a document has a high rank, the document is selected as an offender because it forces the system to make a mistake. The offenders are normally defined as the top-K negative documents in sorted D^- [10]. Given that positive documents are the main source of features, we expect the total number of offenders not more than the positive documents. Therefore, we set $K = \lceil \frac{|D^+|}{3} \rceil$ in our experiments. In the first revision ($loop = 0$), where T contains only positive terms and no negative terms having added yet, the top-j negative documents are omitted for offender selection. The initial features come from positive documents only, and the positive features are more important than negative features at the beginning. An experimental parameter σ is used here and set as $\sigma = \lfloor \frac{|D^-|}{|D^+|} \rfloor$.

To be clear, Table 3 and 4 are used as an example for the selection of offenders process. Table 4 shows a list of ranked negative documents using the terms appearing in Table 3. The first step is to eliminate the documents with weight less than or equal 0. Thus, d_6, d_7 from Table 4 are ignored for offenders. For the sample shown on Table 3 and 4, the number of training documents is 13 with a distribution of $|D^+| = 6$ and $|D^-| = 7$. Therefore, $K = \lceil \frac{6}{3} \rceil = 2$ and if $(loop = 0)$ then $j = \sigma = \lfloor \frac{7}{6} \rfloor = 1$; otherwise, $j = 0$. After that, started from $j + 1$ and counting for K documents, the documents in this range are selected as offenders. As a result d_3, d_4 from Table 4 are selected as offenders at the first loop ($loop = 0$). In the second and third loops the same process is repeated with $j = 0$ and the updated list of terms is used.

Steps 8 and 9 extract negative features (DP^-, T_0) from selected negative documents D_3^-. The ***SPMining***(D_3^-, min_sup) algorithm is employed to discover negative patterns DP^- and T_0, including all terms in patterns of DP^-. Table 5 shows a list of terms extracted from offenders.

Steps 10 to 12 revise the weights for negative specific terms. These steps go three times through a loop with the iteration controlled by Step 13. In each loop, if a specific negative term is extracted at the first time, the algorithm negates its support obtained from the selected negative doc-

Table 4. A set of ranked negative documents with their weight, $|D^-| = 7$.

	Negative documents	*weight*
1	d_1	0.67
2	d_2	0.60
3	d_3	0.44
4	d_4	0.34
5	d_5	0.30
6	d_6	0.00
7	d_7	0.00

Table 5. A set of terms discovered from offender documents.

terms	*weight*
$\langle t_1 \rangle$	-0.20
$\langle t_3 \rangle$	-0.45
$\langle t_7 \rangle$	-0.50
$\langle t_8 \rangle$	-0.75

uments; otherwise, the algorithm cumulates its weight as follows:

$$weight(t) = \alpha \times weight(t, D_3^-) + weight(t).$$

After three loops, the algorithm partitions T into general terms GT and positive specific terms T^+ at Step 14 and 15. It also revises positive specific term weights using the following equation in Step 16 and 17:

$$weight(t) = weight(t) \times (1 + \frac{|\{d|d \in D^+, t \in d\}|}{|D^+|}) \quad (3)$$

Finally, T is updated to include negative specific terms at Step 18.

Table 3 and 5 show a set of terms extracted from positive documents and offenders. The method introduced in Section 2.3 is again used to classify those terms into three main groups: *specific positive*, *specific negative*, and *general* terms:

$$T^+ = \{\langle t_2 \rangle_{0.90}, \langle t_4 \rangle_{0.65}, \langle t_5 \rangle_{0.75}, \langle t_6 \rangle_{0.84}\}$$

$$T^- = \{\langle t_7 \rangle_{-0.50}, \langle t_8 \rangle_{-0.75}\}$$

$$G = \{\langle t_1 \rangle_{(0.34, \cancel{-0.20})}, \langle t_3 \rangle_{(0.65, \cancel{-0.45})}\}$$

The terms in T^+ and T^- have only one weight. However, the terms in general group G have two weights: the first one is for the term occurred in D^+; the second one is for the term occurred in offenders D_3^-. Because the group T^+ is more important than T^- and G, the weight of a $t \in T^+$ is awarded by Eq. (3) based on t's appearance on positive documents. For negative terms T^-, the term weights are updated via a three-loops technique as shown at Step 11. The groups of terms with updated weights are:

$$T^+ = \{\langle t_2 \rangle_{1.8=0.90*(1+\frac{6}{6})}, \langle t_4 \rangle_{1.19}, \langle t_5 \rangle_{1.5}, \langle t_6 \rangle_{1.12}\}$$

$$T^- = \{\langle t_7 \rangle_{-0.50}, \langle t_8 \rangle_{-0.75}\}$$

$$G = \{\langle t_1 \rangle_{0.34}, \langle t_3 \rangle_{0.65}\}$$

NFMining calls three times *SPMining*. The total number of negative documents used in these three times equals $O(|D^+|)$. Therefore, *NFMining* for mining negative patterns has the same complexity as the *SPMining* for mining positive patterns in D^+. *NFMining* also takes times for sorting D^-, assigning weights to terms, and partitioning terms into categories. The time complexity for these operations is $O(|D^-|(log^{|D^-|} + |T|) + |T|^2)$.

3.4 Final Retrieval

Given a topic, the feature terms are extracted by using Algorithm *NFMining* and assigned with a value $weight(t)$, as discussed previously. These features were used in our experiments to perform the final retrieval. Because the volume of ClueWeb09 Category-B corpus is huge, the final retrieval was separated to two steps in order to reduce the complexity.

At the first step, for each topic we retrieved about 30,000 candidate documents based on only title search from the ClueWeb09 Category-B corpus. The process of query expansion (discussed in Section 2.3) was reused here for candidate retrieval. In our investigation on the results of Phase 1 submission, a limitation was exposed that the knowledge specified in the world ontology was not up-to-date. The LCSH system used for ontology construction was the 2006 version. As a result, the ontology missed some up-to-date knowledge, e.g., that about "Obama" and "Obama family tree". In order to solve this problem, at Stage 2 we used world knowledge extracted from the Web using Google API. For each topic, ten Web documents were retrieved and pooled with the training set generated from the instances (library catalog). As discussed in Section 2.3, a set of expanding query terms was then extracted and used for candidate retrieval. Finally, approximately 30,000 candidate documents were retrieved from the Category-B corpus by accumulating the $pr_\beta(t)$ of the terms that occurred in the document titles.

In the next step, we filtered the candidates based on document contents using the features discovered from positive and negative judgements, as discussed previously. The 30,000 candidates were re-ranked by accumulating the $weight(t)$ of features (see Algorithm *NFMining*) that occurred in document contents. After that, the top 1,000 documents were selected and submitted as the final retrieved results against the given topic.

Topic	εMap		StatAP		Score	Topic	εMap		StatAP		Score
1	13	11	***12***	***12***	0.371	26	***5***	***19***	***9***	***16***	0.6557
2	***10***	***13***	***10***	***15***	0.6613	27	16	9	16	9	0.3
3	***4***	***11***	***7***	***9***	0.5676	28	16	9	21	4	0.2459
4	17	7	17	7	0.2982	29	12	3	11	8	0.3235
5	12	4	***9***	***12***	0.4286	30	***10***	***15***	***8***	***17***	0.6508
6	19	5	19	5	0.2545	31	21	4	18	7	0.2419
7	***8***	***15***	***9***	***16***	0.7258	32	19	6	11	14	0.3621
8	***6***	***11***	12	7	0.5	33	***12***	***12***	***10***	***14***	0.5088
9	13	10	***9***	***16***	0.5345	34	16	7	16	8	0.3519
10	15	7	17	6	0.3585	35	***9***	***15***	***7***	***17***	0.5893
11	***9***	***14***	13	11	0.4833	36	11	12	15	8	0.4074
12	21	4	14	11	0.2344	37	15	2	11	7	0.2857
13	11	1	7	6	0.4194	38	***10***	***14***	***7***	***18***	0.5873
14	***12***	***13***	14	11	0.45	39	13	10	***11***	***14***	0.4483
15	***9***	***16***	***12***	***13***	0.5333	40	8	1	12	5	0.2
16	***10***	***10***	***10***	***15***	0.5536	41	***4***	***17***	***7***	***15***	0.6226
17	23	1	19	6	0.1455	42	17	3	7	6	0.25
18	12	11	***6***	***18***	0.6481	43	***9***	***15***	***9***	***16***	0.5517
19	6	0	***0***	***0***	0	44	***9***	***11***	13	11	0.434
20	-	-	-	-	-	45	18	7	***8***	***17***	0.5937
21	***11***	***12***	16	7	0.5085	46	***8***	***15***	***9***	***15***	0.5902
22	***9***	***16***	***9***	***16***	0.7187	47	***8***	***13***	***9***	***13***	0.5357
23	***5***	***11***	11	7	0.5	48	15	4	14	6	0.25
24	***8***	***8***	10	3	0.3421	49	11	9	***10***	***10***	0.3958
25	***10***	***15***	***10***	***15***	0.6562	50	13	8	12	9	0.375
						All	9	16	13	12	0.4844

Table 6. Evaluation of Phase 1 performance

4 Results and Discussions

As discussed previously, the Relevance Feedback track was designed to evaluate a system's capacity of finding quality user relevance feedback and utilizing relevance judgement. In Phase 1, each group submitted five documents for (pseudo) relevance feedback; in Phase 2, groups ran their relevance feedback algorithms based on different sets of judged docs from Phase 1, including their own Phase 1 docs, and several other groups' Phase 1 documents. Evaluation then compared the intrinsic quality of the Phase 1 feedback, as well as each group's relevance feedback algorithm.

Four methods, εMap [1], MapA, P10A, and StatAP [2], were used in the track to measure the performance of Phase 2 runs. εMap and StatAP were applied to the runs using the testing set of only ClueWeb09 Category-B, whereas MapA and P10A were applied to those using the whole ClueWeb09 English set. Because our experiments were based on only ClueWeb09 Category-B, measuring our performance by MapA and P10A might not give us an adequate, substantial analysis. Thus, we investigated our results with only the εMap and StatAP in this discussion.

The quality of a set of Phase 1 extracted documents could be marked if more groups using the set in Phase 2 had better performance than using other Phase 1 sets, when applying to the same relevance feedback algorithm. Table 6 shows the detailed results for the evaluation of our Phase 1 results. In each εMap or StatAP column, the first digit shows the number of runs that using our Phase 1 set was outperformed by using another groups' Phase 1 sets, whereas the second digit shows the number of runs that using ours outperformed using others. Therefore, a larger deviation of two digits indicates higher quality of our pseudo relevance feedback retrieved in Phase 1 when the second digit is greater than the first. In Table 6, those tie or wining comparisons are flagged by the bold, italic font. In terms of εMap performance, using our Phase 1 retrieved feedback was better then (or equal to) using other groups' retrieved feedbacks in 23 out of 49 topics (Topic 20 was dropped because it had no relevant docs). In terms of StatAP, the tie or wining topic number is 24 out of 49. In overall εMap performance of counting 49 topics, the number of runs our Phase 1 set was better than is 16, much more than the number of runs (9) our Phase 1 set was worse than. In overall StatAP performance, the two numbers in the pair is quite close (13 vs. 12). Base on the results, the pseudo relevance feedback retrieved by our group in Phase 1 had a relatively high quality. This is also confirmed by the performance comparisons illustrated in Fig. 4, where our submission (QUT.1) is indexed in a middle position (ahead of 16 groups but behind 13 groups). Out system's capacity of finding quality user relevance feedback is encouraging.

Phase 2 evaluated a system's performance of using relevance judgement for retrieval. The Stage 2 in our model was to use both positive and negative feedback judgements

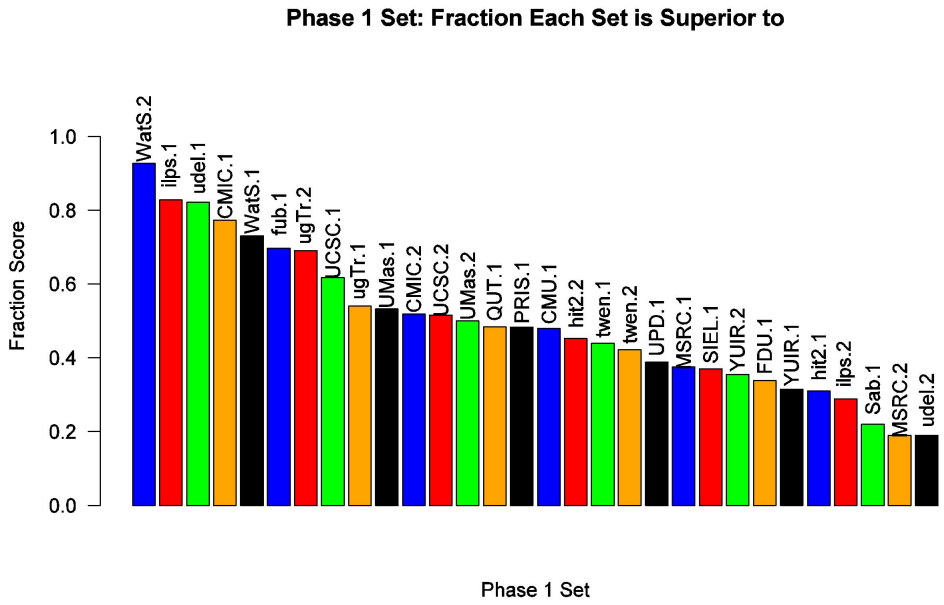


Figure 4. Phase 1 Performance Comparison

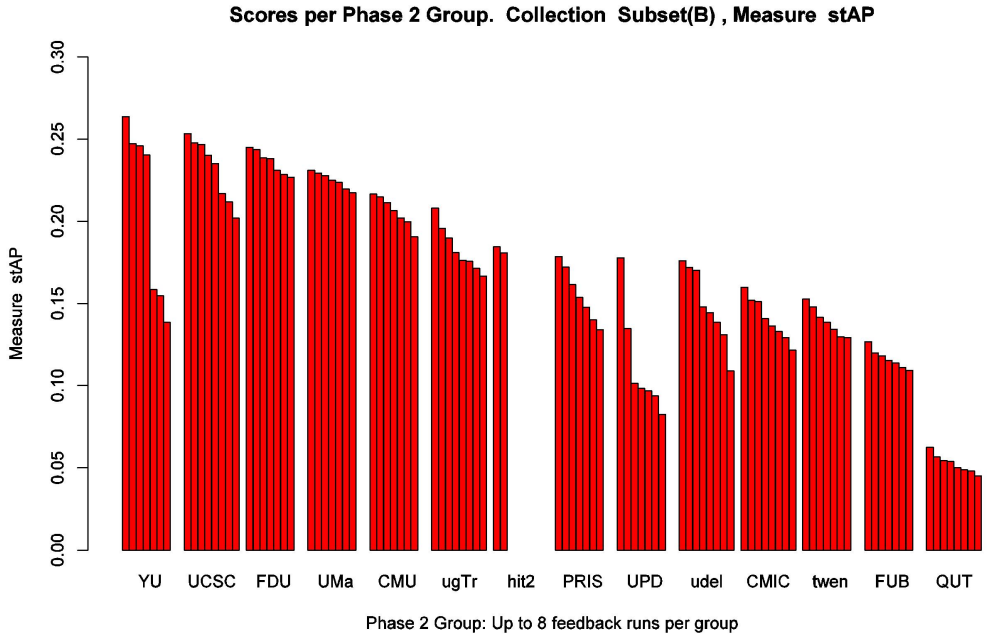


Figure 5. Phase 2 Performance Comparison

for information retrieval. Though many reports suggested that negative relevance judgements were useless or of a little help [4,5,7], this idea has been successfully tested in our previous work [9] on an experimental environment setup by Reuters Corpus Volume 1 (RCV1) corpus [8] and TREC filtering track. The work showed that the method significantly outperformed both the state-of-the-art term-based methods underpinned by Okapi BM25 or Support Vector Machine and pattern based methods on precision, recall and F measures. However, in this track our Phase 2 performance was unsatisfactory, according to the comparison plotted in Fig. 5. In our investigation, we found that the unsatisfactory performance was largely caused by the difficulties encountered when coping with the large testbed, ClueWeb09 Category-B.

Performing content search in ClueWeb09 Category-B for each topic was time and computational resource consuming that we could not afford, according to the track's tough schedule and our accessible resources. ClueWeb09 Category-B is a huge corpus with 1.5 terabyte data, approximate 45,000,000 documents. Pre-processing of ClueWeb09 Category-B required investment of a large amount of time and use of high performance computer. Unfortunately, as the first time in our lab to deal with the High Performance Computer (HPC) Centre in QUT, the poor collaboration and the shortage of HPC experience stole a large amount of our time. As a result, time became against us in the experiments. Consequently, in order to simplify the complexity in maximum with only minimal sacrifice of effectiveness, as discussed in Section 3.4 we separated the Phase 2 search into two steps: for each topic, (i) retrieving about 30,000 candidates from ClueWeb09 Category-B based on only title search; (ii) re-ranking those candidates based on contents and submitting the top 1,000 documents as the final results. We expected with 30,000 candidates we could have only a limited portion of relevant documents missing. However, as shown on Fig. 5, the final result of Phase 2 was disappointing.

The evaluation methods and our Stage 2 method have a basic difference on term weight evaluation. This may also cause the disappointing result in Phase 2. εMap and StatAP are term-based methods that evaluate term weights based on term distribution in documents. Due to the large volume, the ClueWeb09 corpus does not have precise judgements for the testing set (like those manual judgements in RCV1 for topics R101-R150 in TREC 11 Filtering track). In order to test a relevance feedback method, based on term-based algorithms, εMap and StatAP computationally judged the testing set. However, our Stage 2 method is pattern-based. Term weights are evaluated based on term distribution in discovered patterns rather than that in documents (as discussed in Section 3). Therefore, there may exist a problem that the performance of our pattern-based method could be underestimated when using term-based computational judgements to measure. This problem actually happened in our previous experiments: when using RCV1's manual judgements (topics R101-R150), this pattern-based Stage 2 method was largely succeed in the experiments and significantly improved the performance of an information filtering system from using Rocchio, BM25, and SVM [9]; however, such performance improvement became relatively slight when experimented with RCV1's computational judgements (topics R151-R200). Though at this stage it is still too early to justify this problem, it will be interesting to investigate this problem in our future work and test our pattern-based method with more data sets.

5 Conclusion

This paper investigated a model that was experimented in the TREC 2009 Relevance Feedback track. The model had two stages, corresponding to the design of the track. Given a topic, the first stage of our model used a world knowledge ontology to discover user background knowledge for query expansion, and then retrieved the pseudo relevance feedback. From both the positive and negative user relevance judgements, the second stage method mined specific and general features, and used these features to benefit information retrieval. According to the evaluation results, the model performed well in Stage 1 but unsatisfactory in Stage 2. The unsatisfactory performance was caused by the difficulties in coping with the large ClueWeb09 Category-B corpus.

Our participation on this TREC 2010 Relevance Feedback track was an innovative exploration of using both positive and negative feedback judgements in information retrieval. The participation also demonstrated that using a world knowledge ontology is capable of discovering user background knowledge and improving information retrieval. In our future work, further investigation and experiments will be carried on based on full content search on ClueWeb09 Category-B, rather than half title-search half content-search in this reported experiment.

Acknowledgements

The work presented in this paper was partially supported by Grants DP0988007 from the Australian Research Council and NSC98-2218-E-468-002 from the National Science Council of Taiwan.

References

[1] B. Carterette. Robust test collections for retrieval evaluation. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 55–62, New York, NY, USA, 2007. ACM.

[2] B. Carterette, V. Pavlu, E. Kanoulas, J. A. Aslam, and J. Allan. If i had a million queries. In *ECIR '09: Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval*, pages 288–300, Berlin, Heidelberg, 2009. Springer-Verlag.

[3] S. Gauch, J. Chaffee, and A. Pretschner. Ontology-based personalized search and browsing. *Web Intelligence and Agent Systems*, 1(3-4):219–234, 2003.

[4] B. He, C. Macdonald, I. Ounis, J. Peng, and R. L. T. Santos. University of glasgow at trec 2008: Experiments in blog, enterprise, and relevance feedback tracks with terrier. In *TREC*, 2008.

[5] R. Kaptein, J. Kamps, and D. Hiemstra. The impact of positive, negative and topical relevance feedback. In *TREC*, 2008.

[6] L. Khan, D. McLeod, and E. Hovy. Retrieval effectiveness of an ontology-based model for information selection. *The International Journal on Very Large Data Bases*, 13(1):71–85, 2004.

[7] M. Lease. Incorporating relevance and pseudo-relevance feedback in the markov random field model. In *TREC*, 2008.

[8] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li. RCV1: A New Benchmark Collection for Text Categorization Research. *Journal of Machine Learning Research*, 5:361–397, 2004.

[9] Y. Li, A. Algarni, S.-T. Wu, and Y. Xu. Mining negative relevance feedback for information filtering. In *Proceedings of the IEEE/WIC/ACM international conference on Web Intelligence*, pages 606–613, 2009.

[10] Y. Li and N. Zhong. Mining Ontology for Automatically Acquiring Web User Information Needs. *IEEE Transactions on Knowledge and Data Engineering*, 18(4):554–568, 2006.

[11] Y. Li, X. Zhou, P. Bruza, Y. Xu, and R. Y. Lau. A two-stage text mining model for information filtering. In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management*, pages 1023–1032, New York, NY, USA, 2008. ACM.

[12] X. Ling, Q. Mei, C. Zhai, and B. Schatz. Mining multi-faceted overviews of arbitrary topics in a text collection. In *KDD '08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 497–505, New York, NY, USA, 2008. ACM.

[13] S. E. Middleton, N. R. Shadbolt, and D. C. D. Roure. Ontological user profiling in recommender systems. *ACM Transactions on Information Systems (TOIS)*, 22(1):54–88, 2004.

[14] S. E. Robertson and I. Soboroff. The TREC 2002 filtering track report. In *Text REtrieval Conference*, 2002.

[15] S. Scott and S. Matwin. Feature engineering for text classification. In *ICML '99: Proceedings of the Sixteenth International Conference on Machine Learning*, pages 379–388, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[16] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys (CSUR)*, 34(1):1–47, 2002.

[17] A. Sieg, B. Mobasher, and R. Burke. Web search personalization with ontological user profiles. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 525–534, New York, NY, USA, 2007. ACM.

[18] X. Tao, Y. Li, and N. Zhong. A personalized ontology model for web information gathering. *Accepted by IEEE Transaction on Knowledge and Data Engineering*, December 2009.

[19] X. Tao, Y. Li, N. Zhong, and R. Nayak. Ontology mining for personalized web information gathering. In *Proceedings of the 2007 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 351–358, 2007.

[20] S.-T. Wu, Y. Li, and Y. Xu. Deploying approaches for pattern refinement in text mining. In *Proceedings of the Sixth International Conference on Data Mining*, pages 1157–1161, 2006.

[21] S.-T. Wu, Y. Li, Y. Xu, B. Pham, and C. P. Automatic pattern taxonomy exatraction for web mining. In *Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence*, pages 242–248, Beijing, China, 2004.