Mining Negative Relevance Feedback for Information Filtering

Yuefeng Li, Abdulmohsen algarni
School of Information Technology
Queensland University of Technology
Brisbane, Australia
{*y2.li,a1.algarni*}*@qut.edu.au*

Sheng-Tang Wu
Dept. of Information Science and Applications, Asia University
Dublin, Taiwan
swu@asia.edu.tw

Yue Xue
School of Information Technology
Queensland University of Technology
Brisbane, Australia
yue.xu@qut.edu.au

Abstract

It is a big challenge to clearly identify the boundary between positive and negative streams. Several attempts have used negative feedback to solve this challenge; however, there are two issues for using negative relevance feedback to improve the effectiveness of information filtering. The first one is how to select constructive negative samples in order to reduce the space of negative documents. The second issue is how to decide noisy extracted features that should be updated based on the selected negative samples. This paper proposes a pattern mining based approach to select some offenders from the negative documents, where an offender can be used to reduce the side effects of noisy features. It also classifies extracted features (i.e., terms) into three categories: positive specific terms, general terms, and negative specific terms. In this way, multiple revising strategies can be used to update extracted features. An iterative learning algorithm is also proposed to implement this approach on RCV1, and substantial experiments show that the proposed approach achieves encouraging performance.

1. Introduction

Traditional IF models were developed based on a term-based user profile approach (see [11], [13], [8]). The advantage of term-based profiles is efficient computational performance as well as mature theories for term weighting, which have emerged over the last couple of decades from the information retrieval (IR) and machine learning communities. However, term-based profiles suffer from the problems of polysemy and synonymy. As IF systems are sensitive to data sets, it is still a challenging issue to significantly improve the effectiveness of IF systems.

To overcome the limitations of term based approaches, data mining (pattern mining) based techniques have been used for information filtering since patterns are more discriminative and arguably carry more "semantics" than terms. One special filtering task was to extract usage patterns from Web logs [3], [26]. Another promising techniques were pattern taxonomy models (PTM) [19], [21] that discovered closed sequential patterns in text documents, where a pattern was a set of terms that frequently appeared in paragraph.

Pattern based approaches have shown encouraging improvements on effectiveness. However, two challenging issues have arisen when pattern mining techniques were introduced for IF systems. The first one is how to deal with low frequency patterns because the measures used from data mining (e.g., "support" and "confidences") to learn the patterns turn out be not suitable in the filtering stage [8]. The second issue is how to effectively use negative feedback to revise extracted features (including patterns and terms) for information filtering.

Many people believe that there are plenty negative information available and negative documents are very useful because they can help users to search for accurate information [20]. However, whether negative feedback can indeed largely improve filtering accuracy is still an open question. The existing methods of using negative feedback for IF can be grouped into two approaches. The first approach is to revise terms that appear in both positive samples and negative samples (e.g., Rocchio based models and SVM [13] based filtering models). This heuristics is obvious when people assume that terms are isolated atoms. The second approach is based on how often terms appear or do not appear in positive samples and negative samples (e.g., probabilistic models [1], and BM25 [13]). However, usually people view terms in multiple perspectives when they attempt to find what they want. They normally use two dimensions ("specificity" and "exhaustivity") for deciding the relevance of documents, paragraphes or terms. For example, "JDK" is a specific term for "Java Language", and "LIB" is more general than "JDK" because it is also frequently used for C and C++.

Based on this observation, this paper proposes a pattern mining based approach for using negative feedback. It firstly extracts an initial list of terms from positive documents and selects some constructive negative documents (or called offenders). It then extracts terms from negative patterns in selected negative documents. It also classifies all terms into three categories: the positive specific terms, general terms, and negative specific terms. In this way, multiple revising strategies are used for terms in different categories. In the implementation, it recommends to increment positive specific terms' weights only and declines negative specific terms' weights based on their occurrences in discovered negative patterns. Substantial experiments show that the proposed approach achieves exciting performance.

The remainder of this paper is organized as follows. Section 2 introduces a detailed overview of the related works. Section 3 reviews the concepts of pattern taxonomy mining. Section 4 introduces the equations for evaluating term weights based on discovered patterns. Section 5 describes the proposed method of using negative feedback. The empirical results and discussion are reported in Section 6, and the last section describes concluding remarks.

2. Related Work

Different from IR systems, IF systems were commonly personalized to support long-term information needs of users [2]. The main distinction between IR and IF was that IR systems used "queries" but IF systems used "user profiles". The tasks of the filtering included adaptive filtering, and batch and routing filtering. Adaptive filtering involves feedback to dynamically adapt IF systems [5] [23]. In this paper, the focus is on the breakthrough for batch and routing filtering.

Normally, IF systems tended to learn a map $rank : D \rightarrow R$ such that $rank(d)$ corresponded to the relevance of a document d, where D denoted a set of documents, $\mathbb{R}$ was the set of real numbers. In [11], $rank$ was divided into two functions, such that $rank = f_1 \circ f_2$, where f_1 ($f_1 : D \rightarrow \{C_1, \ldots, C_m\}$), and f_2 ($f_2 : \{C_1, \ldots, C_m\} \rightarrow R$) were maps, respectively; and $C_1, C_2, \ldots, C_m$ were clusters. This method used a set of clusters based on a kind of classification method, e.g., the neural network [10].

The aim of the filtering track in TREC [13] was to measure the ability of IF systems to build profiles using sets of training documents to separate relevant and non-relevant documents. The basic term-based IF models used in TREC 2002 were SVM, Rocchio's algorithm, probabilistic models, and BM25.

Term-based IF models have been developed recently which took into consideration more constraints in relation to the labeled data in training sets; for instance, Rocchio-style classifiers [6]; ranking SVM [12]; and BM25 for structured documents [15]. However, the research on term-based IF models has arguably hit somewhat of a wall in terms of effectiveness improvement possibly due to the ambiguity problem mentioned earlier.

To overcome the disadvantages of term-based approaches, sequential patterns and closed patterns have been developed in pattern taxonomy models (PTM) [19], [21]. These approaches introduced data mining techniques to information filtering; however, too many noisy patterns adversely affect PTM systems [8]. The major research issue is how to use negative feedback to significantly reduce the effects of noisy patterns. Traditional data mining techniques can only achieve a little progress for the effectiveness because they can only discuss this problem at the pattern level. This paper starts to consider human being's perspective about relevance and uses a two dimension concept to classify terms into three groups: positive specific terms, general terms and negative specific terms by using the two classes of training documents. In this perspective, term weights can be evaluated accurately based on their appearances in both positive patterns and negative patterns.

Table 1. A set of paragraphs

$Parapgraph$	$Terms$
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

3. Pattern Taxonomy Mining

In this paper, we assume that all documents are split in paragraphs. So a given document d yields a set of paragraphs $PS(d)$. Let D be a training set of documents, which consists of a set of positive documents, D^+; and a set of negative documents, D^-. Let $T = \{t_1, t_2, \ldots, t_m\}$ be a set of terms (or keywords) which are extracted from the set of positive documents, D^+.

3.1. Frequent and Closed Patterns

Given a *termset* X, a set of terms, in document d, $\ulcorner X \urcorner$ is used to denote the covering set of X for d, which includes all paragraphs $dp \in PS(d)$ such that $X \subseteq dp$, i.e., $\ulcorner X \urcorner = \{dp | dp \in PS(d), X \subseteq dp\}$. Its *absolute support* is the number of occurrences of X in $PS(d)$, that is $sup_a(X) = |\ulcorner X \urcorner|$. Its *relative support* is the fraction of the paragraphs that contain the pattern, that is, $sup_r(X) = \frac{|\ulcorner X \urcorner|}{|PS(d)|}$. A termset X is called *frequent pattern* if its sup_a (or sup_r) $\geq min\_sup$, a minimum support.

Table 1 lists a set of paragraphs for a given document d, where $PS(d) = \{dp_1, dp_2, \ldots, dp_6\}$, and duplicate terms are removed. Let $min\_sup = 3$ giving rise to ten frequent patterns which are illustrated in Table 2. Normally not all frequent patterns are useful [19], [22]. For example, pattern $\{t_3, t_4\}$ always occurs with term t_6 in paragraphs (see Table 1); therefore, we want to keep the larger pattern only.

Given a termset X, its covering set $\ulcorner X \urcorner$ is a subset of paragraphs. Similarly, given a set of paragraphs $Y \subseteq PS(d)$, we can define its *termset*, which satisfies

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

The closure of X is defined as follows:

$$Cls(X) = termset(\ulcorner X \urcorner).$$

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

Let X be a closed pattern. We have

$$sup_a(X_1) < sup_a(X) \quad (1)$$

for all pattern $X_1 \supset X$.

Table 2. Frequent patterns and covering sets

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

3.2. Pattern Taxonomy

Patterns can be structured into a taxonomy by using the *is-a* (or *subset*) relation and closed patterns. For example, Table 2 contains ten frequent patterns; however, it includes only three closed patterns: $< t_3, t_4, t_6 >$, $< t_1, t_2 >$, and $< t_6 >$. Simply, a pattern taxonomy is described as a set of pattern-absolute support pairs, for example PT=$\{\langle t_3, t_4, t_6\rangle_3$, $\langle t_1, t_2\rangle_3$, $\langle t_6\rangle_5\}$, where non-closed patterns are pruned. After pruning, some direct "is-a" retaliations may be changed, for example, pattern $\{t_6\}$ would become a direct sub-pattern of $\{t_3, t_4, t_6\}$ after pruning non-closed patterns $< t_3, t_6 >$ and $< t_4, t_6 >$.

Smaller patterns in the taxonomy, for example pattern $\{t_6\}$, are usually more general because they could be used frequently in both positive and negative documents; and larger patterns, for example pattern $\{t_3, t_4, t_6\}$, in the taxonomy are usually more specific since they may only used in positive documents.

3.3. Closed Sequential Patterns

A sequential pattern $s =< t_1, \ldots, t_r >$ ($t_i \in T$) is an ordered list of terms. A sequence $s_1 =< x_1, \ldots, x_i >$ is a sub-sequence of another sequence $s_2 =< y_1, \ldots, y_j >$, denoted by $s_1 \sqsubseteq s_2$, 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 usually say s_1 is a sub-pattern of s_2, and s_2 is a super-pattern of s_1. In the following, we simply say patterns for sequential patterns.

Given a pattern (an ordered *termset*) X in document d, $\ulcorner X \urcorner$ is still used to denote the covering set of X, which includes all paragraphs $ps \in PS(d)$ such that $X \sqsubseteq ps$, i.e., $\ulcorner X \urcorner = \{ps | ps \in PS(d), X \sqsubseteq ps\}$. Its *absolute support* and *relative support* are defined as the same as for the normal patterns.

A sequential pattern X is called *frequent pattern* if its relative support $\geq min\_sup$, a minimum support. The property of closed patterns (see Eq. (1)) can be used to define closed sequential patterns. A frequent sequential pattern X is called *closed* if not $\exists$ any super-pattern X_1 of X such that $sup_a(X_1) = sup_a(X)$.

4. Deploying Patterns on Terms

The evaluation of term supports (weights) in this paper is different from the term-based approaches. For a term based approach, the evaluation of a given term's weight is based on its appearance in documents. For pattern mining, terms are weighted according to their appearance in discovered patterns.

To improve the efficiency of the pattern taxonomy mining, an algorithm, *SPMining*(D^+, min_sup) [19], was proposed (also used in [21], [8]) to find closed sequential patterns for all document $\in D^+$, which used the well-known *Apriori* property in order to reduce the searching space. For all positive document $d \in D^+$, the *SPMining* algorithm discovered all closed sequential patterns based on a given min_sup.

Let SP_1, SP_2, ..., $SP_{|D^+|}$ be the sets of discovered closed sequential patterns for all document $d_i \in D^+ (i = 1, \cdots, |D^+|)$. For a given term, its support in these discovered patterns can be described as follows:

$$support(t, D^+) = \sum_{i=1}^{|D^+|} \frac{|\{p | p \in SP_i, t \in p\}|}{\sum_{p \in SP_i} |p|}$$

After the supports of terms have been computed from the training set, the following rank will be assigned to an incoming document d that can be used to decide its relevance:

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

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

5. Mining Negative Feedback

In general, the concept of relevance is subjective; and normally people can describe the relevance of a topic (or document) in two dimensions: the specificity and exhaustivity, where "specificity" describes the extent to which the topic focuses on what users want, and "exhaustivity" describes the extent to which the topic discusses what users want. It is easy for human being to do so. However, it is very difficult to use the two dimensions for IF systems. In this section, we first discuss how to use the two dimensions for understanding the different roles of the selected terms. We also presents an algorithm for both negative document selection and term weight revision.

5.1. Specific and General Terms

Formally, let DP^+ be the union of all discovered patterns of pattern taxonomies of D^+, and DP^- be the union of all discovered negative patterns of pattern taxonomies of D^-, where a closed sequential pattern of D^- is called negative pattern. Given a term $t \in T$, its *exhaustivity* is the number of discovered patterns in both DP^+ and DP^- that contain t, and its *specificity* is the number of discovered patterns in DP^+ but not in DP^- that contain t. Based on this understanding,

in this paper we classify terms into three groups. We call a term a *general term* if it appear in both patterns discovered in positive documents and negative patterns discovered in negative documents. We also call terms positive (or negative) *specific terms* if they appear only in patterns discovered in positive (or negative) documents only.

Based on the above discussion, we have the following definitions for the set of general terms GT, the set of positive specific terms T^+, and the set of negative specific terms T^-:

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

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

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

It is easy to verify that $GT \cap T^+ \cap T^- = \emptyset$. Therefore, GT, T^+ and T^- is a partition of all terms in discovered patterns and negative patterns.

To present user profiles for a given topic, normally we believe that specific terms are very useful for the topic in order to distinguish to other topics. However, some experimental result show that using only specific terms are not enough to improve the performance of information filtering because user information needs cannot simply be covered by documents that only contain the specific terms. Therefore, the best way is to use the specific terms mixed with some of the general terms.

5.2. Strategies of Revision

After we can classify terms into three categories, we firstly show the basic process of revising discovered features in the training set. This process can help readers to understand the proposed strategies for revising discovered features in different categories.

The process first extracts initial features in the positive documents in the training set, which include terms and patterns. It then selects some negative samples (or called offenders) in the set of negative documents in the training set. It also extracts negative features, including both terms and negative patterns, from the selected negative documents using the same pattern mining technique as used for the feature extraction in positive documents. In addition, it revises the initial features and obtains revised features. The process can be repeated for several times as follows: selecting negative documents, extracting negative features and revising revised features.

Algorithm *NFMining*(D) describes the details of the strategies of the revision, where we assume 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, $< T, DP^+, DP^- >$, have been extracted from positive documents D^+ before we start the algorithm, where we let $DP^- = \emptyset$. We also let the experimental parameter $\alpha = -1$ that will be used for calculating weights of terms in negative patterns.

Step 1 initializes the set of general terms GT, the set of positive specific terms T^+ and the set of negative specific terms T^-, where $loop$ is used to control the times of the revision. Step 2 and 3 calculate terms' weights for all term in T.

Step 4 and 5 rank documents in the set of negative documents, where if t is a negative specific term, its weight is the revising weight calculating in step 10 and 11. The weight function can be described as follows:

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

Step 6 and 7 sort the negative documents based on documents' rank values, and select offenders, some negative documents. If a document's rank less than or equals to 0 that means this document is clearly negative to the system. A document has hight rank that means the document is an offender because it forces the system make mistake. The offenders are normally defined as the top-K negative documents in sorted D^- [7]. In this paper, we let $K = \lceil \frac{|D^+|}{3} \rceil$. In the first revision ($loop = 0$), we ignore the top-j negative documents for offender selection since the initial features only coming from positive documents and we believe that positive features are more important than negative features in the beginning, where $j = \lfloor \frac{|D^-|}{|D^+|} \rfloor$, the largest integer that less than or equals to $\frac{|D^-|}{|D^+|}$.

Step 8 and 9 extract negative features (DP^-, T_0) from selected negative documents D_3^-, where it calls algorithm *SPMining*(D_3^-, min_sup) to discover negative patterns DP^-, and T_0 that includes all terms in patterns in DP^-.

Step 10 to 12 revise negative specific terms' weights. These steps will go through a loop for three times and the iteration is controlled by step 13. In each loop, when a specific negative term is extracted in the first time, the algorithm simply negatives its support obtained from the selected negative documents; otherwise, the algorithm cumulates its weight as follows:

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

After three loops, the algorithm participates T into general terms GT and positive specific terms T^+ in step 14 and 15. It also revises positive specific terms' weights using the following equation in step 16 and 17:

$$weight(t) = weight(t) + weight(t) * (\frac{|\{d|d \in D^+, t \in d\}|}{|D^+|})$$

At last, it updates T to include negative specific terms in step 18.

NFMining calls three times *SPMining* and the total negative documents used in the three times is $O(|D^+|)$; therefore, it takes the same computation time for mining patterns in selected negative documents as the *SPMining* does for mining patterns in positive documents. *NFMining* also takes times for sorting D^-, assigning weights to terms and partitioning terms into groups. The time complexity for these operations is $O(|D^-|(log^{(|D^-|)} + |T|) + |T|^2)$.

NFMining(D)

Input: A training set, $\{D^+, D^-\}$, parameter $\alpha = -1$;
extracted features $< T, DP^+, DP^- >$, $DP^- = \emptyset$;
support function and minimum support min_sup.
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: $weight(t) = support(t, D^+)$;
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 ranking order,
let $j = \lfloor \frac{|D^-|}{|D^+|} \rfloor$ if $loop = 0$, otherwise $j = 0$;
7: $D_3^- = \{d_i | d_i \in D^-, j \leq i < \lceil \frac{|D^+|}{3} \rceil + j\}$;
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** $weight(t) = \alpha \times support(t, D_3^-)$
else $weight(t) = \alpha \times support(t, D_3^-) + weight(t)$;
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: $weight(t) = weight(t) + weight(t) * (\frac{|\{d | d \in D^+, t \in d\}|}{|D^+|})$;
18: $T = T \cup T^-$;

6. Evaluation

In this section, we first discuss the data collection used for our experiments. We also describe the baseline models and their implementation. In addition, we present the experimental results and the discussion.

6.1. Data

Reuters Corpus Volume 1 (RCV1) was used to test the effectiveness of the proposed model. RCV1 corpus consists of all and only English language stories produced by Reuter's journalists between August 20, 1996, and August 19, 1997 with total 806,791 documents. The document collection is divided into training sets and test sets.

TREC (2002) has developed and provided 100 topics for the filtering track aiming at building a robust filtering system. The topics are of two types: 1) A first set of 50 topics are developed by the assessors of the National Institute of Standards and Technology (NIST)(i.e., assessor topics); The relevance judgements have been made by assessor of NIST. 2) A second set of 50 topics have been constructed artificially from intersections of pairs of Reuters categories (i.e., intersection topics) [18]. For that reason we use the 50 assessor topics in this paper where the result is more reliable.

RCV1 collection is marked in XML. To avoid bias in experiments, all of the meta-data information in the collection have been ignored. The documents are treated as plain text documents by preprocessing the documents. The tasks of removing stop-words according to a given stop-words list and stemming term by applying the Porter Stemming algorithm are conducted [9].

6.2. Baseline Models and Setting

Four baseline models are used: the classic Rocchio model, a BM25 based IF model, a SVM based model, and PTM model. In this paper, our new model is called Negative PaTtern Mining model (N-PTM).

The Rocchio algorithm [16] has been widely adopted in the areas of text categorization and information filtering. It can be used to build the profile for representing the concept of a topic which consists of a set of relevant (positive) and irrelevant (negative) documents. The Centroid $\vec{c}$ of a topic can be generated as follows:

$$\alpha \frac{1}{|D^+|} \sum_{\overrightarrow{d} \in D^+} \frac{\overrightarrow{d}}{||\overrightarrow{d}||} - \beta \frac{1}{|D^-|} \sum_{\overrightarrow{d} \in D^-} \frac{\overrightarrow{d}}{||\overrightarrow{d}||}$$

where we set $\alpha = \beta = 1.0$ in this paper.

BM25 [4], [14] is the one of state-of-the-art retrieval functions used in document retrieval. The term weights are estimated using the following BM25 based equation:

$$W(t) = \frac{tf \cdot (k_1 + 1)}{k_1 \cdot ((1-b) + b\frac{DL}{AVDL}) + tf} \cdot \log \frac{\frac{(r+0.5)}{(n-r+0.5)}}{\frac{(R-r+0.5)}{(N-n-R+r+0.5)}}$$

where N is the total number of documents in the training set; R is the number of positive documents in the training set; n is the number of documents which contain term t; r is the number of positive documents which contain term t; tf is the term frequency; DL and *AVDL* are the document length and average document length, respectively; and k_1 and b are the experimental parameters (the values of k_1 and b are set as 1.2 and 0.75, respectively, in this paper).

Information filtering can also be regarded as a special instance of text classification [17]. SVM is a statistical method that can be used to find a hyperplane that best separates two classes. SVM achieved the best performance on the Reuters-21578 data collection for document classification [24]. The decision function in SVM is defined as:

$$h(x) = \sin(w \cdot x + b) = \begin{cases} +1 & \text{if } (w \cdot x + b) > 0 \\ -1 & \text{otherwise} \end{cases}$$

where x is the input object; b in $\Re$ is a threshold and $w = \sum_{i=1}^{l} y_i \alpha_i x_i$ for the given training data: $(x_i, y_i), ..., (x_l, y_l)$, where $x_i \ \varepsilon \ \Re^n$ and y_i equals $+1$ (-1), if document x_i is labeled positive (negative). $\alpha_i \ \epsilon \ \Re$ is the weight of the training example x_i and satisfies the following constraints:

$$\forall_i : \alpha_i \geqslant 0 \quad \text{and} \quad \sum_{i=1}^{l} \alpha_i y_i = 0 \tag{2}$$

To compare with other baseline models, we tried to use SVM to rank documents rather than to make binary decisions.

Table 3. Results in all assessor topics for PTM and N-PTM

	PTM	N-PTM	$\%chg$	p-value
b/p	0.4299932	0.4684968	+8.95	0.001974031
MAP	0.4435398	0.4871910	+9.84	0.001415044
IAP	0.4641946	0.50668984	+9.15	0.002250558
$F_{\beta=1}$	0.439174956	0.4637	+5.58	0.000829231

For this purpose, threshold b can be ignored. We also believe that the positive documents in the training set should have the same importance to user information needs because the training set was only simply divided into positive documents and negative documents. So we assign the same α_i value (i.e., 1) to each positive document first, and then determine the same α_i (i.e., $\acute{\alpha}$) value to each negative document based on Eq. (2). Therefore, we use the following weighting function to estimate the similarity between a testing document and a given topic:

$$weight(d) = w \cdot d$$

where $\cdot$ means *inner product*; d is the term vector of the testing document; and

$$w = (\sum_{d_i \in D^+} d_i) + (\sum_{d_j \in D^-} d_j \acute{\alpha}).$$

For each topic, we also choose 150 terms in the positive documents based on *tf*idf* values for all term-based baseline models.

PTM model is also selected as one of the baselines models because we want to verify that mining negative feedback can significantly improve the performance of PTM. The size of the term set T is 4000 for PTM. We also set $min\_sup = 0.2$ (relative support) for both PTM and N-PTM.

6.3. Results

The effectiveness was measured by four different means: The F-beta (F_β) measure, Mean Average Precision (MAP), the break-even point (b/p), and Interpolated Average Precision (IAP) on *11-points*.

F_β is calculated by the following function:

$$F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}$$

The parameter $\beta = 1$ is used in our study, which means that recall and precision is weighed equally. Mean Average precision is calculated by measuring precision at each relevant document first, and averaging precision over all topics. The *b/p* break-even point indicates the value at which precision equals recall. The larger a *b/p*, MAP, IAP or F_β-measure score is, the better the system performs. *11-points* measure is also used to compare the performance of different systems by averaging precisions at 11 standard recall levels (i.e., recall=0.0, 0.1, ..., 1.0).

Statistical method is also used to analyze the experimental results. The t-test assesses whether the means of two groups

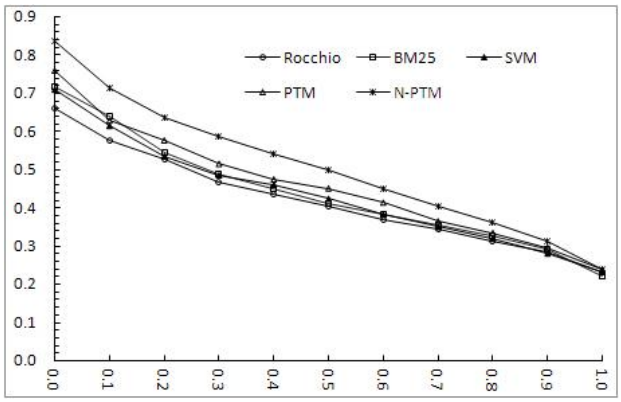

Figure 1. comparison between the proposed method and other approachs.

are statistically different from each other. The paired two-tailed t-test is used in this paper. If DIF represents the difference between observations, the hypotheses are: Ho : $DIF = 0$ (the difference between the two observations is 0). $Ha : DIF \neq 0$ (the difference is not 0). N is the sample size of group. The test statistic is t with $N-1$ degrees of freedom (*df*). If the p-value associated with t is low ($<$0.05), there is evidence to reject the null hypothesis. Thus, there is evidence that the difference in means across the paired observations is significant. The N-PTM model is compared with PTM, Rocchio, BM25, and SVM models for each variable b/p, MAP, IAP, $F_{\beta=1}$ over all the 50 topics, respectively.

6.3.1. PTM(Positive) vs N-PTM(Positive and Negative). In order to see the effectiveness of using both positive and negative feedback in pattern mining approach, in this section we compare the proposed mining negative feedback model (N-PTM) to PTM model which uses positive documents only. The results on the 50 topics for the comparison between PTM and N-PTM over the standard measures are shown in Tables 3. The results of *11-points* on 50 topics are reported in Figure 1.

As shown in Table 3 and Figure 1, the performance of N-PTM model is extremely better than use positive feedback only. Table 3 also shows the percentage changes and the p vales. The statistic results indicate that the proposed mining negative feedback model is extremely statistically significant. Therefore, we conclude that mining negative relevance feedback for information filtering is an exciting achievement for pattern based approaches.

6.3.2. N-PTM vs Term-Based State-of-the-Art Models. The proposed method is also compared with term-based baseline models including Rocchio, BM25, and SVM. The experimental results on all assessor topics are reported in Table 5. The results of *11-points* on all assessor topics are reported in Figure 1.

As shown in Table 5 and Figure 1, the proposed new model (N-PTM) has achieved the best performance results for the

Table 4. Extracted Features in First Ten Topics, where $min\_sup = 0.2$.

Topic	Number of Documents in Training Sets			Number of Extracted Terms				Weight of Extracted Terms			
	Positive	*Negative*	*Offenders*	T^+	T^-	GT	T	$w(T^+)$	$w(T^-)$	$w(GT)$	total weight
101	7	16	5	64	142	35	241	20.2	−34.2	16.6	2.6
102	135	64	21	604	79	221	904	250.4	−152.3	663.2	761.3
103	14	50	7	106	434	51	591	24.8	−47.2	25.3	2.9
104	120	74	25	425	123	297	845	131.7	−179.7	678.0	630.0
105	16	21	9	145	144	43	332	76.1	−109.7	50.5	16.9
106	4	40	2	91	48	12	151	22.1	−24.4	11.8	9.5
107	3	58	3	54	544	35	633	10.4	−11.0	7.5	6.9
108	3	50	3	57	77	21	155	16.0	−14.6	10.1	11.4
109	20	20	6	215	53	87	355	63.0	−78.6	99.7	84.2
110	5	86	3	40	103	17	160	11.3	−17.2	19.2	13.4
Total	327	479	84	1801	1747	819	4367	626.0	−668.8	1581.9	1539.2
Avg	33	48	8.4	180	175	82	437	62.6	−66.9	158.2	153.9

Table 5. Results in all assessor topicss, where $\%chg$ is the percentage change over the best term based model.

	Rocchio	BM25	SVM	N-PTM	$\%chg$
b/p	0.420	0.403	0.409	0.468	11.52
MAP	0.430	0.417	0.409	0.487	13.17
IAP	0.452	0.439	0.434	0.507	12.03
$F_{\beta=1}$	0.430	0.421	0.421	0.464	7.88

assessor topics. The improvements are consistent and very significant on the all above measures.

6.4. Discussion

The main process of the proposed approach consists of two steps: offender selection, and the revision of term weights. It is obvious that not all negative feedback are suitable to be selected as offenders, where offenders are the most useful negative documents that can help to balance the percentages of general terms and specific terms in the extracted features. Informally, the documents that have high weight are called offenders. Figure 2 shows the difference between using all negative documents and using offenders in all assessor topics. This figure illustrates that the proposed method for offender selection meets the design objectives.

Table 4 shows the numbers of positive documents, negative documents and offenders in the training sets of the first ten documents (because of the limitation of the paper length, we have not shown all assessor topics here). It also illustrates that only 17% negative documents are selected as offenders, that is, the proposed method is much efficient for reducing the space of negative documents.

For the revision of term weights, the proposed method first classifies extracted terms into general terms and specific terms that is a distinguish advantage comparing with others [20], [25]. The normal belief is that specific terms are more interesting than general terms for a given topic. Therefore, the proposed method only increases the weights of specific terms when it conduces the revision using negative documents.

General terms are not only frequently appear in positive documents, but also frequently appear in some negative documents because negative documents may describe some extent

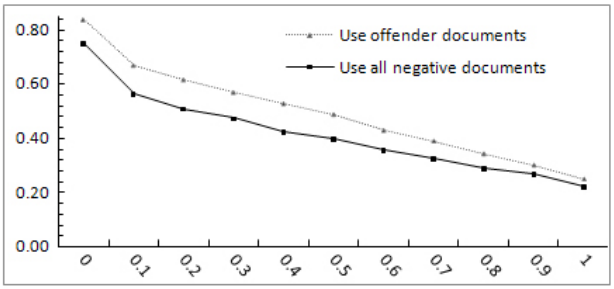

Figure 2. comparison between used all negative documents and used the offender one.

to which the topic discusses what users want. To reduce the side effects of using general terms in the extracted features, the proposed method adds negative specific terms into the extracted features.

Table 4 also shows the numbers of extracted general terms, specific terms and negative specific terms, and their weights. Before revision, it can be seen that more than $72\% = \frac{158.2}{158.2+62.6}$ weights are distributed to general terms although the percentage of general terms is $31\% = \frac{82}{82+180}$ for all extracted terms in positive documents.

After revision, 1747 negative specific terms are added into T for the ten topics in Table 4, and they are assigned weight -66.9 in average. In this way, these negative specific terms could reduce the side effects of general terms if both general terms and negative specific terms appear in negative documents because now only 59% weights could be distributed to general terms considering positive specific terms get weight 62.2 in average and general terms get $158.2-66.9$ in average.

Comparing with the best model of the state-of-the-art models, the proposed approach achieves excellent performance with 11.15% (max 13.17% and min 7.88%) average percentage change for all 4 measures.

7. Conclusions

Negative documents are very useful for information filtering. However, whether negative feedback can largely improve

filtering accuracy is still an open question. This paper presents a pattern mining based approach for this open question. It introduces a method to select negative documents (or called offenders) that are close to the extracted features. It also proposes an approach to classify extracted terms into three groups: positive specific terms, general terms and negative specific terms. In this perspective, it presents an iterative algorithm to revise extracted features. Compared with the state-of-the-art models, the results of experiments on RCV1 collection demonstrate that the effectiveness of information filtering can be significantly improved by the proposed new approach. This research provides a promising methodology for evaluating term weights based on discovered patterns (rather than documents) in both positive and negative documents.

References

[1] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, 1999.

[2] N. J. Belkin and W. B. Croft. Information filtering and information retrieval: two sides of the same coin? *Commun. ACM*, 35(12):29–38, 1992.

[3] X. Fu, J. Budzik, and K. J. Hammond. Mining navigation history for recommendation. In *IUI '00: Proceedings of the 5th international conference on Intelligent user interfaces*, pages 106–112, New York, NY, USA, 2000. ACM.

[4] K. S. Jones, S. Walker, and S. E. Robertson. A probabilistic model of information retrieval: development and comparative experiments - part 1. *Inf. Process. Manage.*, 36(6):779–808, 2000.

[5] R. Y. K. Lau, P. Bruza, and D. Song. Belief revision for adaptive information retrieval. In *SIGIR*, pages 130–137, 2004.

[6] X. Li and B. Liu. Learning to classify texts using positive and unlabeled data. In *IJCAI*, pages 587–594, 2003.

[7] 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.

[8] 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, Napa Valley, California, USA, 2008.

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

[10] J. Mostafa and W. Lam. Automatic classification using supervised learning in a medical document filtering application. *Inf. Process. Manage.*, 36(3):415–444, 2000.

[11] J. Mostafa, S. Mukhopadhyay, W. Lam, and M. J. Palakal. A multilevel approach to intelligent information filtering: Model, system, and evaluation. *ACM Trans. Inf. Syst.*, 15(4):368–399, 1997.

[12] T. Qin, X.-D. Zhang, D.-S. Wang, T.-Y. Liu, W. Lai, and H. Li. Ranking with multiple hyperplanes. In *SIGIR*, pages 279–286, 2007.

[13] S. E. Robertson and I. Soboroff. The trec 2002 filtering track report. In *TREC*, 2002.

[14] S. E. Robertson, S. Walker, and M. Hancock-Beaulieu. Okapi at trec-7: Automatic ad hoc, filtering, vlc and interactive. In *TREC*, pages 199–210, 1998.

[15] S. E. Robertson, H. Zaragoza, and M. J. Taylor. Simple bm25 extension to multiple weighted fields. In *CIKM*, pages 42–49, 2004.

[16] J. Rocchio. *Relevance feedback in information retrieval*, volume In The SMART Retrieval System: Experiments in Automatic Document Processing. Prentice Hall, 1971.

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

[18] I. Soboroff and S. Robertson. Building a filtering test collection for trec 2002. In *SIGIR*, pages 243–250, New York, NY, USA, 2003. ACM.

[19] S.T.Wu, Y.Li, Y. Xu, B. Pham, and P.Chen. Automatic pattern-taxonomy extraction for web mining. In *the IEEE/WIC/ACM International Conference on Web Intelligence*, pages 242 – 248, China, 2004.

[20] X. Wang, H. Fang, and C. Zhai. A study of methods for negative relevance feedback. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 219–226, New York, NY, USA, 2008. ACM.

[21] S.-T. Wu, Y. Li, and Y. Xu. Deploying approaches for pattern refinement in text mining. In *ICDM*, pages 1157–1161, 2006.

[22] Y. Xu and Y. Li. Generating concise association rules. In *CIKM*, pages 781–790, 2007.

[23] Y. Yang, A. Lad, N. Lao, A. Harpale, B. Kisiel, and M. Rogati. Utility-based information distillation over temporally sequenced documents. In *SIGIR*, pages 31–38, 2007.

[24] Y. Yang and X. Liu. A re-examination of text categorization methods. In *SIGIR*, pages 42–49, 1999.

[25] Y. Zhang. Using bayesian priors to combine classifiers for adaptive filtering. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 345–352, New York, NY, USA, 2004. ACM.

[26] Y. Zhang and J. Callan. Combining multiple forms of evidence while filtering. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 587–595, Morristown, NJ, USA, 2005. Association for Computational Linguistics.