Effective Pattern Discovery for Text Mining

Ning Zhong, *Senior Member, IEEE,* Yuefeng Li, *Member, IEEE,* Sheng-Tang Wu, *Member, IEEE*

Abstract—Many data mining techniques have been proposed for mining useful patterns in text documents. However, how to effectively use and update discovered patterns is still an open research issue, especially in the domain of text mining. Since most existing text mining methods adopted term-based approaches, they all suffer from the problems of polysemy and synonymy. Over the years, people have often held the hypothesis that pattern (or phrase) based approaches should perform better than the term-based ones, but many experiments do not support this hypothesis. This paper presents an innovative and effective pattern discovery technique which includes the processes of pattern deploying and pattern evolving, to improve the effectiveness of using and updating discovered patterns for finding relevant and interesting information. Substantial experiments on RCV1 data collection and TREC topics demonstrate that the proposed solution achieves encouraging performance.

Index Terms—Text mining, pattern mining, pattern evolving, information filtering.

1 Introduction

Due to the rapid growth of digital data made available in recent years, knowledge discovery and data mining have attracted a great deal of attention with an imminent need for turning such data into useful information and knowledge. Many applications, such as market analysis and business management, can benefit by the use of the information and knowledge extracted from a large amount of data. Knowledge discovery can be viewed as the process of nontrivial extraction of information from large databases, information that is implicitly presented in the data, previously unknown and potentially useful for users. Data mining is therefore an essential step in the process of knowledge discovery in databases.

In the past decade, a significant number of data mining techniques have been presented in order to perform different knowledge tasks. These techniques include association rule mining, frequent itemset mining, sequential pattern mining, maximum pattern mining and closed pattern mining. Most of them are proposed for the purpose of developing efficient mining algorithms to find particular patterns within a reasonable and acceptable time frame. With a large number of patterns generated by using data mining approaches, how to effectively use and update these patterns is still an open research issue. In this paper, we focus on the development of a knowledge discovery model to effectively use and update the discovered patterns and apply it to the field of text mining.

Text mining is the discovery of interesting knowledge in text documents. It is a challenging issue to find accurate knowledge (or features) in text documents to help users to find what they want. In the beginning, Information Retrieval (IR) provided many term-based methods to solve this challenge, such as Rocchio and probabilistic models [4], rough set models [23], BM25 and SVM [34] based filtering models. The advantages of term-based methods include efficient computational performance as well as mature theories for term weighting, which have emerged over the last couple of decades from the IR and machine learning communities. However, term-based methods suffer from the the problems of polysemy and synonymy, where polysemy means a word has multiple meanings, and synonymy is multiple words having the same meaning. The semantic meaning of many discovered terms is uncertain for answering what users want.

Over the years, people have often held the hypothesis that phrase-based approaches could perform better than the term-based ones, as phrases may carry more "semantics" like information. This hypothesis has not fared too well in the history of IR [19], [40], [41]. Although phrases are less ambiguous and more discriminative than individual terms, the likely reasons for the discouraging performance include: (*i*) phrases have inferior statistical properties to terms, (*ii*) they have low frequency of occurrence, and (*iii*) there are large number of redundant and noisy phrases among them [41].

In the presence of these set backs, sequential patterns used in data mining community have turned out to be a promising alternative to phrases [13], [50] because sequential patterns enjoy good statistical properties like terms. To overcome the disadvantages of phrase-based approaches, pattern mining based

- *N. Zhong is with the International WIC Institute, Beijing University of Technology, China, and the Department of Life Science and Informatics, Maebashi Institute of Technology, Japan.*
  E-mail: zhong@maebashi-it.ac.jp
- *Y. Li is with the Discipline of Computer Science, Queensland University of Technology, Brisbane, QLD 4001, Australia.*
  E-mail: y2.li@qut.edu.au
- *S.-T. Wu is with the Department of Information Science and Applications, Asia University, Taiwan 41354.*
  E-mail: swu@asia.edu.tw

approaches (or pattern taxonomy models (PTM) [50], [51]) have been proposed, which adopted the concept of closed sequential patterns, and pruned non-closed patterns. These pattern mining based approaches have shown certain extent improvements on the effectiveness. However, the *paradox* is that people think pattern-based approaches could be a significant alternative, but consequently less significant improvements are made for the effectiveness compared with term-based methods.

There are two fundamental issues regarding the effectiveness of pattern-based approaches: low-frequency and misinterpretation. Given a specified topic, a highly frequent pattern (normally a short pattern with large support) is usually a general pattern, or a specific pattern of low frequency. If we decrease the minimum support, there are a lot of noisy patterns would be discovered. Misinterpretation means the measures used in pattern mining (e.g., "support" and "confidence") turn out to be not suitable in using discovered patterns to answer what users want. The difficult problem hence is how to use discovered patterns to accurately evaluate the weights of useful features (knowledge) in text documents.

Over the years, IR has developed many mature techniques which demonstrated that terms were important features in text documents. However, many terms with larger weights (e.g., the term frequency and inverse document frequency (tf*idf) weighting scheme) are general terms because they can be frequently used in both relevant and irrelevant information. For example, term "LIB" may have larger weight than "JDK" in a certain of data collection; but we believe that term "JDK" is more specific than term "LIB" for describing "Java Programming Language"; and term "LIB" is more general than term "JDK" because term "LIB" is also frequently used in C and C++. Therefore, it is not adequate for evaluating the weights of the terms based on their distributions in documents for a given topic, although this evaluating method has been frequently used in developing IR models.

In order to solve the above *paradox*, this paper presents an effective pattern discovery technique, which first calculates discovered specificities of patterns and then evaluates term weights according to the distribution of terms in the discovered patterns rather than the distribution in documents for solving the misinterpretation problem. It also considers the influence of patterns from the negative training examples to find ambiguous (noisy) patterns and try to reduce their influence for the low-frequency problem. The process of updating ambiguous patterns can be referred as pattern evolution. The proposed approach can improve the accuracy of evaluating term weights because discovered patterns are more specific than whole documents.

We also conduct numerous experiments on the latest data collection, Reuters Corpus Volume 1 (RCV1) and TREC (Text Retrieval Conference) filtering topics, to evaluate the proposed technique. The results show that the proposed technique outperforms up-to-date data mining-based methods, concept-based models and the state-of-the-art term-based methods.

The rest of this paper is structured as follows. Section 2 discusses related work. Section III provides some definitions about closed patterns, PTM and closed sequential patterns. Sections IV and Section V propose the techniques of pattern deploying and inner pattern evolution in PTM, respectively. Section VI presents experimental setting and results for evaluating the proposed approach. Finally, Section 7 gives concluding remarks.

2 Related Work

Many types of text representations have been proposed in the past. A well-known one is the bag of words that uses keywords (terms) as elements in the vector of the feature space. In [21], the tf*idf weighting scheme is used for text representation in Rocchio classifiers. In addition to TFIDF, the global IDF and entropy weighting scheme is proposed in [9] and improves performance by an average of 30%. Various weighting schemes for the bag of words representation approach were given in [1], [14], [38]. The problem of the bag of words approach is how to select a limited number of features among an enormous set of words or terms in order to increase the system's efficiency and avoid *overfitting* [41]. In order to reduce the number of features, many dimensionality reduction approaches have been conducted by the use of feature selection techniques, such as Information Gain, Mutual Information, Chi-Square, Odds ratio, and so on. Details of these selection functions were stated in [19], [41].

The choice of a representation depended on what one regards as the meaningful units of text and the meaningful natural language rules for the combination of these units [41]. With respect to the representation of the content of documents, some research works have used phrases rather than individual words. In [7], the combination of *unigram* and *bi-grams* was chosen for document indexing in text categorization (TC) and evaluated on a variety of feature evaluation functions (FEF). A phrase-based text representation for Web document management was also proposed in [44].

In [3], data mining techniques have been used for text analysis by extracting co-occurring terms as descriptive phrases from document collections. However, the effectiveness of the text mining systems using phrases as text representation showed no significant improvement. The likely reason was that a phrase-based method had "lower consistency of assignment and lower document frequency for terms" as mentioned in [18].

Term-based ontology mining methods also provided some thoughts for text representations. For example, hierarchical clustering [28], [29] was used to determine synonymy and hyponymy relations between keywords. Also, the pattern evolution technique was introduced in [25] in order to improve the performance of term-based ontology mining.

Pattern mining has been extensively studied in data mining communities for many years. A variety of efficient algorithms such as Apriori-like algorithms [2], [31], [49], PrefixSpan [32], [53], FP-tree [10], [11], SPADE [56], SLPMiner [42] and GST [12] have been proposed. These research works have mainly focused on developing efficient mining algorithms for discovering patterns from a large data collection. However, searching for useful and interesting patterns and rules was still an open problem [22], [24], [52]. In the field of text mining, pattern mining techniques can be used to find various text patterns, such as sequential patterns, frequent itemsets, co-occurring terms and multiple grams, for building up a representation with these new types of features. Nevertheless, the challenging issue is how to effectively deal with the large amount of discovered patterns.

For the challenging issue, closed sequential patterns has been used for text mining in [51], which proposed that the concept of closed patterns in text mining was useful and had the potential for improving the performance of text mining. Pattern taxonomy model was also developed in [50], [51] to improve the effectiveness by effectively using closed patterns in text mining. In addition, a two stages model that used both term-based methods and pattern-based methods was introduced in [26] to significantly improve the performance of information filtering.

Natural language processing (NLP) is a modern computational technology that can help people to understand the meaning of text documents. For a long time, NLP was struggling for dealing with uncertainties in human languages. Recently, a new concept-based model [45], [46] was presented to bridge the gap between NLP and text mining, which analyzed terms on the sentence and document levels. This model included three components. The first component analyzed the semantic structure of sentences; the second component constructed a conceptual ontological graph (COG) to describe the sematic structures; and the last component extracted top concepts based on the first two components to build feature vectors using the standard vector space model. The advantage of the concept-based model is that it can effectively discriminate between non-important terms and meaningful terms which describe a sentence meaning. Compared with the above methods, the concept-based model usually relies upon its employed NLP techniques.

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

TABLE 2
Frequent patterns and covering sets

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

3 PATTERN TAXONOMY MODEL

In this paper, we assume that all documents are split into 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 $T = \{t_1, t_2, \dots, t_m\}$ be a set of terms (or keywords) which can be extracted from the set of positive documents, D^+.

3.1 Frequent and Closed Patterns

Given a termset X 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_r (or sup_a) $\geq min\_sup$, a minimum support.

Table 1 lists a set of paragraphs for a given document d, where $PS(d) = \{dp_1, dp_2, \dots, dp_6\}$, and duplicate terms were removed. Let $min\_sup = 50\%$, we can obtain ten frequent patterns in Table 1 using the above definitions. Table 2 illustrates the ten frequent patterns and their covering sets.

Not all frequent patterns in Table 2 are useful. For example, pattern $\{t_3, t_4\}$ always occurs with term t_6 in paragraphs, i.e., the shorter pattern, $\{t_3, t_4\}$, is always a part of the larger pattern, $\{t_3, t_4, t_6\}$, in all of the paragraphs. Hence, we believe that the shorter one, $\{t_3, t_4\}$, is a noise pattern and expect to keep the larger pattern, $\{t_3, t_4, t_6\}$, only.

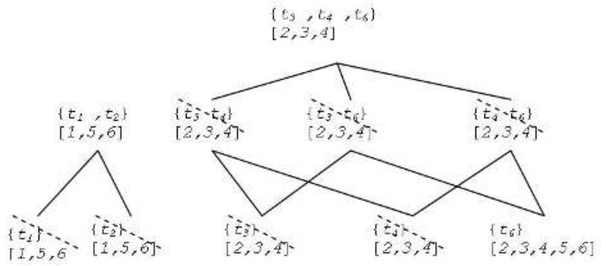

Fig. 1. Pattern taxonomy

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

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

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 can prove that

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

for all patterns $X_1 \supset X$; otherwise, if $sup_a(X_1) = sup_a(X)$, we have

$$\ulcorner X_1 \urcorner = \ulcorner X \urcorner,$$

where $sup_a(X_1)$ and $sup_a(X)$ are the absolute support of pattern X_1 and X, respectively.

We also have

$$Cls(X) = termset(\ulcorner X \urcorner) = termset(\ulcorner X_1 \urcorner) \supseteq X_1 \supset X,$$

that is, $Cls(X) \neq X$.

3.2 Pattern Taxonomy

Patterns can be structured into a taxonomy by using the *is-a* (or *subset*) relation. For the example of Table 1, where we have illustrated a set of paragraphs of a document, and the discovered 10 frequent patterns in Table 2 if assuming $min\_sup = 50\%$. There are, however, only three closed patterns in this example. They are $< t_3, t_4, t_6 >$, $< t_1, t_2 >$, and $< t_6 >$.

Figure 1 illustrates an example of the pattern taxonomy for the frequent patterns in Table 2, where the nodes represent frequent patterns and their covering sets; non-closed patterns can be pruned; the edges are "is-a" relation. 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.

Smaller patterns in the taxonomy, for example pattern $\{t_6\}$, (see Fig. 1) 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. The semantic information will be used in the pattern taxonomy to improve the performance of using closed patterns in text mining, which will be further discussed in the next section.

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 $\exists j_1, \ldots, j_y$ such that $1 \leq j_1 < j_2 \ldots < j_y \leq j$ and $x_1 = y_{j_1}, x_2 = y_{j_2}, \ldots, x_i = y_{j_y}$. 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* 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 sequential pattern X is called frequent pattern if its relative support (or absolute 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 Pattern Deploying Method

In order to use the semantic information in the pattern taxonomy to improve the performance of closed patterns in text mining, we need to interpret discovered patterns by summarizing them as d-patterns (see the definition below) in order to accurately evaluate term weights (supports). The rational behind this motivation is that d-patterns include more semantic meaning than terms that selected based on a term-based technique (e.g., tf*idf). As a result, a term with higher tf*idf value could be meaningless if it has not cited by some d-patterns (some important parts in documents). The evaluation of term weights (supports) is different to the normal term-based approaches. In the term based approaches, the evaluation of term weights are based on the distribution of terms in documents. In this research, terms are weighted according to their appearances in discovered closed patterns.

4.1 Representations of Closed Patterns

It is complicated to derive a method to apply discovered patterns in text documents for information filtering systems. To simplify this process, we firstly review the composition operation $\oplus$ defined in [25].

Let p_1 and p_2 be sets of term-number pairs. $p_1 \oplus p_2$ is called the *composition* of p_1 and p_2 which satisfies:

$$p_1 \oplus p_2 = \{(t, x_1 + x_2) | (t, x_1) \in p_1, (t, x_2) \in p_2\} \bigcup \{(t, x) | (t, x) \in p_1 \cup p_2, not((t, \_) \in p_1 \cap p_2)\},$$

where _ is the wild card that matches any number.

For the special case we have $p \oplus \emptyset = p$; and the operands of the composition operation are interchangeable. The result of the composition is still a set of term-number pairs.

For example,

$$\{(t_1, 1), (t_2, 2), (t_3, 3)\} \oplus \{(t_2, 4)\} = \{(t_1, 1), (t_2, 6), (t_3, 3)\}$$

or

$$\{(t_1, 2\%), (t_2, 5\%), (t_3, 9\%)\} \oplus \{(t_1, 1\%), (t_2, 3\%)\}$$

$$= \{(t_1, 3\%), (t_2, 8\%), (t_3, 9\%)\}.$$

Formally, for all positive documents $d_i \in D^+$, we first deploy its closed patterns on a common set of terms T in order to obtain the following *d-patterns* (deployed patterns, non sequential weighted patterns):

$$\widehat{d_i} = \{(t_{i_1}, n_{i_1}), (t_{i_2}, n_{i_2}), \ldots, (t_{i_m}, n_{i_m})\} \qquad (2)$$

where t_{i_j} in pair (t_{i_j}, n_{i_j}) denotes a single term and n_{i_j} is its *support* in d_i which is the total absolute supports given by closed patterns that contain t_{i_j}; or n_{i_j} (simply in this paper) is the total number of closed patterns that contain t_{i_j}.

For example, using Fig. 1 and Table 1, we have
$sup_a(< t_3, t_4, t_6 >) = 3$,
$sup_a(< t_1, t_2 >) = 3$,
$sup_a(< t_6 >) = 5$, and
$\widehat{d} = \{(t_1, 3), (t_2, 3), (t_3, 3), (t_4, 3), (t_6, 8)\}$.

The process of calculating d-patterns can be easily described by using the $\oplus$ operation in Algorithm 1 (PTM) that will be described in the next sub-section, where a *term's support* is the total number of closed patterns that contain the term.

Table 3 illustrates a real example of pattern taxonomy for a set of positive documents.

We also can obtain the d-patterns of the five sample documents in Table 3 which are expressed as follows:

$$\begin{aligned}
\widehat{d_1} &= \{(\textit{carbon}, 2), (\textit{emiss}, 1), (\textit{air}, 1), (\textit{pollut}, 1)\} \\
\widehat{d_2} &= \{(\textit{greenhous}, 1), (\textit{global}, 2), (\textit{emiss}, 1)\} \\
\widehat{d_3} &= \{(\textit{greenhous}, 1), (\textit{global}, 1), (\textit{emiss}, 1)\} \\
\widehat{d_4} &= \{(\textit{carbon}, 1), (\textit{air}, 2), (\textit{antarct}, 1)\} \\
\widehat{d_5} &= \{(\textit{emiss}, 1), (\textit{global}, 1), (\textit{pollut}, 1)\}.
\end{aligned}$$

Let DP be a set of d-patterns in D^+, and $p \in DP$ be a d-pattern. We call $p(t)$ the absolute support of term t, which is the numbers of patterns that contain t in the corresponding patterns taxonomies. In order to effectively deploy patterns in different taxonomies from the different positive documents, d-patterns will be normalized using the following assignment sentence:

$$p(t) \longleftarrow p(t) \times \frac{1}{\sum_{t \in T} p(t)}.$$

Actually the relationship between d-patterns and terms can be explicitly described as the following *association mapping* [25], a set-value function:

$$\beta : DP \rightarrow 2^{T \times [0,1]}, \qquad (3)$$

such that

$$\beta(p_i) = \{(t_1, w_1), (t_2, w_2), \ldots, (t_k, w_k)\},$$

for all $p_i = \{(t_1, f_1), (t_2, f_2), \ldots, (t_k, f_k)\} \in DP$, where $T = \{t | (t, f) \in p, p \in DP\}$, and

$$w_i = \frac{f_i}{\sum_{j=1}^{k} f_j}.$$

$\beta(p_i)$ is called the *normal form* (or normalized d-pattern) of d-pattern p_i in this paper, and

$$termset(p_i) = \{t_1, t_2, \ldots, t_k\}.$$

TABLE 3
Example of a set of positive documents consisting of pattern taxonomies. The number beside each sequential pattern indicates the absolute support of pattern.

Doc.	Pattern taxonomies	Sequential patterns
d_1	$PT_{(1,1)}$	$\{\langle carbon \rangle_4, \langle carbon, emiss \rangle_3\}$
	$PT_{(1,2)}$	$\{\langle air, pollut \rangle_2\}$
d_2	$PT_{(2,1)}$	$\{\langle greenhous, global \rangle_3\}$
	$PT_{(2,2)}$	$\{\langle emiss, global \rangle_2\}$
d_3	$PT_{(3,1)}$	$\{\langle greenhous \rangle_2\}$
	$PT_{(3,2)}$	$\{\langle global, emiss \rangle_2\}$
d_4	$PT_{(4,1)}$	$\{\langle carbon \rangle_3\}$
	$PT_{(4,2)}$	$\{\langle air \rangle_3, \langle air, antarct \rangle_2\}$
d_5	$PT_{(5,1)}$	$\{\langle emiss, global, pollut \rangle_2\}$

4.2 D-Pattern Mining Algorithm

To improve the efficiency of the pattern taxonomy mining, an algorithm, *SPMining*, was proposed in [50] to find all closed sequential patterns, which used the well-known *Apriori* property in order to reduce the searching space.

Algorithm 1 (PTM) describes the training process of finding the set of d-patterns. For every positive document, the SPMining algorithm is first called in step (4) giving rise to a set of closed sequential patterns SP. The main focus of this paper is the deploying process, which consists of the d-pattern discovery and term support evaluation. In Algorithm 1, All discovered patterns in a positive document are composed into a d-pattern giving rise to a set of d-patterns DP in step (6) to step (9). Thereafter from step (12) to step

(19), term supports are calculated based on the normal forms for all terms in d-patterns.

Let $m = |T|$ be the number of terms in T, $n = |D^+|$ be the number of positive documents in a training set, K be the average number of discovered patterns in a positive document, and k be the average number of terms in a discovered pattern. We also assume that the basic operation is a comparison between two terms.

The time complexity of the d-pattern discovery (from step (6) to step (9)) is $O(Kk^2n)$. Step 10 takes $O(mn)$. Step 12 also gets all terms from d-patterns and takes $O(m^2n^2)$. Step 13 to step 15 initialize support function and take $O(m)$, and the step 16 to step 20 takes $O(mn)$. Therefore, the time complexity of pattern deploying is

$$O(Kk^2n + mn + m^2n^2 + m + mn) = O(Kk^2n + m^2n^2).$$

```
input : positive documents $D^+$; minimum support, $min\_sup$.
output: d-patterns $DP$, and supports of terms.
1  $DP = \emptyset$;
2  foreach document $d \in D^+$ do
3      let $PS(d)$ be the set of paragraphs in $d$;
4      $SP$ = SPMining($PS(d)$, $min\_sup$);
5      $\widehat{d} = \emptyset$;
6      foreach pattern $p_i \in SP$ do
7          $p = \{(t, 1) | t \in p_i\}$;
8          $\widehat{d} = \widehat{d} \oplus p$;
9      end
10     $DP = DP \cup \{\widehat{d}\}$;
11 end
12 $T = \{t | (t, f) \in p, p \in DP\}$;
13 foreach term $t \in T$ do
14     $support(t) = 0$;
15 end
16 foreach d-pattern $p \in DP$ do
17     foreach $(t, w) \in \beta(p)$ do
18         $support(t) = support(t) + w$;
19     end
20 end
```

Algorithm 1: PTM(D^+, min_sup)

After the supports of terms have been computed from the training set, the following weight will be assigned to all incoming documents d for deciding its relevance:

$$weight(d) = \sum_{t \in T} support(t) \tau(t, d), \tag{4}$$

where $support(t)$ is defined in Algorithm 1; and $\tau(t, d) = 1$ if $t \in d$; otherwise $\tau(t, d) = 0$.

5 Inner Pattern Evolution

In this section, we discuss how to re-shuffle supports of terms within normal forms of d-patterns based on negative documents in the training set. The technique will be useful to reduce the side-effects of noisy patterns because of the low-frequency problem. This technique is called inner pattern evolution (IPE) here, because it only changes a pattern's term supports within the pattern.

A threshold is usually used to decide the relevance of incoming documents. Using the d-patterns, the threshold can be defined naturally as follows:

$$Threshold(DP) = \min_{p \in DP} \Big(\sum_{(t,w) \in \beta(p)} support(t) \Big). \tag{5}$$

A noise negative document nd in D^- is a negative document that the system falsely identified as a positive, that is $weight(nd) \geq Threshold(DP)$. In order to reduce the noise, we need to track which d-patterns have been used to give rise to such error. We call these patterns offenders of nd.

An offender of nd is a d-pattern that has at least one term in nd. The set of offenders of nd is defined by:

$$\Delta(nd) = \{p \in DP | termset(p) \cap nd \neq \emptyset\}. \tag{6}$$

There are two types of offenders: (1) a complete conflict offender which is a subset of nd; and (2) a partial conflict offender which contains part of terms of nd.

The basic idea of updating patterns is explained as follows. Complete conflict offenders are removed from d-patterns firstly. For partial conflict offenders, their term supports are reshuffled in order to reduce the effects of noise documents.

```
input : a training set $D = D^+ \cup D^-$; a set of
        d-patterns $DP$; and an experimental coefficient
        $\mu$.
output: a set of term-support pairs $np$.
1  $np \leftarrow \emptyset$;
2  $threshold = Threshold(DP)$;// see Eq. (5)
3  foreach noise negative document $nd \in D^-$ do
4      if $weight(nd) \geq threshold$ then
       $\Delta(nd) = \{p \in DP | termset(p) \cap nd \neq \emptyset\}$;
5      $NDP = \{\beta(p) | p \in DP\}$;
6      Shuffling($nd$, $\Delta(nd)$, $NDP$, $\mu$. $NDP$); //call Alg. 3
7      foreach $p \in NDP$ do
8          $np \leftarrow np \oplus p$;
9      end
10 end
```

Algorithm 2: IPEvolving(D^+, D^-, DP, μ)

The main process of inner pattern evolution is implemented by the algorithm IPEvolving (see Algorithm 2). The inputs of this algorithm are a set of d-patterns DP, a training set $D = D^+ \cup D^-$. The output is a composed d-pattern. Step (2) in IPEvolving is used to estimate the threshold for finding the noise negative documents. Step 3 to Step 10 revise term supports by using all noise negative documents. Step (4) is to find noise documents and the corresponding offenders. Step (5) gets normal forms of d-patterns NDP. Step (6) calls algorithm Shuffling (see Algorithm 3) to update NDP according to noise documents. Steps (7) to (9) compose updated normal forms together.

The time complexity of Algorithm 2 is decided by step 2, the number of calls for Shuffling algorithm and the number of using $\oplus$ operation. Step 2 takes $O(nm)$.

For each noise negative pattern nd, the algorithm gets its offenders that takes $O(nm \times |nd|)$ in step 4, and then calls once Shuffling. After that, it calls n $\oplus$ operation that takes $O(nmm) = O(nm^2)$.

The task of algorithm Shuffling is to tune the support distribution of terms within a d-pattern. A different strategy is dedicated in this algorithm for each type of offender. As stated in step (2) in the algorithm Shuffling, complete conflict offenders (d-patterns) are removed since all elements within the d-patterns are held by the negative documents indicating that they can be discarded for preventing interference from these possible "noises".

```
input : a noise document $nd$, its offenders $\Delta(nd)$,
        normal forms of d-patterns $NDP$, and an
        experimental coefficient $\mu$.
output: updated normal forms of d-patterns $NDP$.
1  foreach d-pattern $p$ in $\Delta(nd)$ do
2      if $termset(p) \subseteq nd$ then $NDP = NDP - \{\beta(p)\}$;
       //remove complete conflict offenders
3      else //partial conflict offender
4          offering =
           $(1 - \frac{1}{\mu}) \times \sum_{t \in (termset(p) \cap nd)} support(t)$;
           base = $\sum_{t \in (termset(p) - nd)} support(t)$;
5
6          foreach term $t$ in $termset(p)$ do
7              if $t \in nd$ then
               $support(t) = (\frac{1}{\mu}) \times support(t)$; //shrink
8              else //grow supports
9                  $support(t) =$
                   $support(t) \times (1 + offering \div base)$;
10
11         end
12
13 end
```
Algorithm 3: Shuffling(nd, $\Delta(nd)$, NDP, μ. NDP)

The parameter *offering* is used in step (4) for the purpose of temporarily storing the reduced supports of some terms in a partial conflict offender. The *offering* is part of the sum of supports of terms in a d-pattern where these terms also appear in a noise document. The algorithm calculates the *base* in step (5) which is certainly not zero since $termset(p) - nd \neq \emptyset$; and then updates the support distributions of terms in step (6).

For example, for the following d-pattern

$$\widehat{d} = \{(t_1, 3), (t_2, 3), (t_3, 3), (t_4, 3), (t_6, 8)\}.$$

Its normal form is

$$\{(t_1, 3/20), (t_2, 3/20), (t_3, 3/20), (t_4, 3/20), (t_6, 2/5)\}.$$

Assume $nd = \{t_1, t_2, t_6, t_9\}$, $\widehat{d}$ will be a partial conflict offender since

$$termset(\widehat{d}) \cap nd = \{t_1, t_2, t_6\} \neq \emptyset.$$

Let $\mu = 2$, *offering* $= \frac{1}{2} \times (\frac{3}{20} + \frac{3}{20} + \frac{2}{5}) = \frac{7}{20}$, and *base* $= \frac{3}{20} + \frac{3}{20} = \frac{3}{10}$. Hence, we can get the following updated normal form by using algorithm Shuffling:

$$\{(t_1, 3/40), (t_2, 3/40), (\mathbf{t}_3, 13/40), (\mathbf{t}_4, 13/40), (t_6, 1/5)\}.$$

Let $m = |T|$, $n = |D^+|$ the number of positive documents in a training set, and q be the number of noise negative documents in D^-. The time complexity of algorithm Shuffling is decided by steps 6 to 9. For a given noise negative document nd, its time complexity is $O(nm^2)$ if let $nd = nd \cap T$, where $T = \{t \in termset(p) | p \in DP\}$. Hence, the time complexity of algorithm Shuffling is $O(nm^2)$ for a given noise negative document.

Based on the above analysis about Algorithm 2 and Algorithm 3, the total time complexity of the inner pattern evolution is $O(nm + q(nm|nd| + nm^2) + nm^2)$ = $O(qnm^2)$ considering that the noise negative document nd can be replaced by $nd \cap T$ before conducting the pattern evolution.

The proposed model includes two phases: the training phase and the testing phase. In the training phase, the proposed model first calls Algorithm PTM(D^+, min_sup) to find d-patterns in positive documents (D^+) based on a min_sup, and evaluates term supports by deploying d-patterns to terms. It also calls Algorithm IPEvolving(D^+, D^-, DP, μ) to revise term supports using noise negative documents in D^- based on an experimental coefficient μ. In the testing phase, it evaluates weights for all incoming documents using Eq. (4). The incoming documents then can be sorted based on these weights.

6 Evaluation and Discussion

In this study Reuters text collection is used to evaluate the proposed approach. Term stemming and stopword removal techniques are used in the prior stage of text preprocessing. Several common measures are then applied for performance evaluation and our results are compared with the-state-of-art approaches in data mining, concept-based and term-based methods.

6.1 Experimental Dataset

The most popular used dataset currently is Reuters Corpus Volume 1 (RCV1), which includes 806,791 news articles for the period between 20 August 1996 and 19 August 1997. These documents were formatted by using a structured XML schema. TREC filtering track has developed and provided two groups of topics (100 in total) for RCV1 [37]. The first group includes 50 topics that were composed by human assessors and the second group also includes 50 topics that were constructed artificially from intersections topics. Each topic divided documents into two parts: the training set and the testing set. The training set has a total amount of 5,127 articles and the testing set contains 37,556 articles. Documents in both sets are assigned either positive or negative, where "positive" means the document is relevant to the assigned topic; otherwise "negative" will be shown.

All experimental models use "title" and "text" of XML documents only. The content in "title" is viewed as a paragraph as the one in "text" which consists of paragraphs. For dimensionality reduction, stopword removal is applied and the Porter algorithm [33] is selected for suffix stripping. Terms with term frequency equaling to one are discarded.

6.2 Measures

Several standard measures based on precision and recall are used. The precision is the fraction of retrieved documents that are relevant to the topic, and the recall is the fraction of relevant documents that have been retrieved.

The precision of first *K* returned documents *top-K* is also adopted in this paper. The value of *K* we use in the experiments is 20. In addition, the breakeven point (b/p) is used to provide another measurement for performance evaluation. It indicates the point where the value of precision equals to the value of recall for a topic. The higher the figure of b/p, the more effective the system is. The b/p measure has been frequently used in common information retrieval evaluations.

In order to assess the effect involving both precision and recall, another criterion that can be used for experimental evaluation is *F_β-measure* [20], which combines precision and recall and can be defined by the following equation:

$$F_\beta - measure = \frac{(\beta^2 + 1) * \mathit{precision} * \mathit{recall}}{\beta^2 * \mathit{precision} + \mathit{recall}} \quad (7)$$

where β is a parameter giving weights of precision and recall and can be viewed as the relative degree of importance attributed to precision and recall [41]. A value $\beta = 1$ is adopted in our experiments meaning that it attributes equal importance to precision and recall. When $\beta = 1$, the measure is expressed as:

$$F_1 = \frac{2 * \mathit{precision} * \mathit{recall}}{\mathit{precision} + \mathit{recall}}. \quad (8)$$

The value of $F_{\beta=1}$ is equivalent to the b/p when precision equals to recall. However, the b/p cannot be compared directly to the $F_{\beta=1}$ value since the latter is given a higher score than that of the former [54]. It has also been stated in [30] that the $F_{\beta=1}$ measure is greater or equal to the value of b/p.

Both the b/p and *F_β-measure* are the single-valued measures in that they only use a figure to reflect the performance over all the documents. However, we need more figures to evaluate the system as a whole. Hence, another measure, Interpolated Average Precision (IAP) is introduced and has been adopted before in several research works [17], [43], [54]. This measure is 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). The *11-points* measure is used in our comparison tables indicating the first value of 11 points where recall equals to zero. Moreover, Mean Average Precision (MAP) is used in our evaluation which is calculated by measuring precision at each relevance document first, and averaging precisions over all topics.

6.3 Baseline Models

In order to make a comprehensive evaluation, we choose three classes of models as the baseline models. The first class includes several data mining based methods that we have introduced in Section III. In the following, we introduce other two classes: the concept-based model and term-based methods.

6.3.1 Concept Based Models

A new concept-based model was presented in [45], [46], which analyzed terms on both sentence and document levels. This model used a verb-argument structure which split a sentence into verbs and their arguments. For example, "John hits the ball", where "hits" is a verb, and "John" or "the ball" are the arguments of "hits". Arguments can be further assigned labels such as subjects or objects (or theme). Therefore, a term can be extended and to be either an argument or a verb, and a concept is a labeled term.

For a document d, $tf(c)$ is the number of occurrences of concept c in d; and $ctf(c)$ is called the conceptual term frequency of concept c in a sentence s, which is the number of occurrences of concept c in the verb-argument structure of sentence s. Given a concept c, its tf and ctf can be normalized as $tf_{weight}(c)$ and $ctf_{weight}(c)$, and its weight can be evaluated as follows:

$$weight(c) = tf_{weight}(c) + ctf_{weight}(c)$$

To have a uniform representation, in this paper, we call a concept as a concept-pattern which is a set of terms. For example, verb "hits" is denoted as $\{hits\}$ and its argument "the ball" is denoted as $\{the, ball\}$.

It is complicated to construct a COG. Also, up to now, we have not found any work for constructing COG for describing semantic structures for a set of documents rather than for an individual document for information filtering. In order to give a comprehensive evaluation for comparing the proposed model with the concept-based model, in this paper, we design a concept-based model (CBM) for describing the features in a set of positive documents, which consists of two steps. The first step is to find all of the concepts in the positive documents of the training set, where verbs are extracted from PropBank data set at $http : //verbs.colorado.edu/verb - index/propbank - 1.0.tar.gz$. The second step is to use the deploying approach to evaluate the weights of terms based on their appearances in these discovery concepts. Unlike the proposed model, which uses 4000 features at most, the concept-based model uses all features for each topic. Let CP_i be the set of concepts in $d_i \in D^+$. To synthesize both tf and ctf of concepts in all positive documents, we use the following equation to evaluate term weights

$$W(t) = \sum_{i=1}^{|D^+|} \frac{|\{c|c \in CP_i, t \in c\}|}{\sum_{c \in CP_i} |c|} \tag{9}$$

for all $t \in T$.

We also designed another kind of the concept-based model, called CBM Pattern Matching, which evaluates a document d's relevance by accumulating the weights of concepts that appear in d as follows:

$$weight(d) = \sum_{c \in d} weight(c). \tag{10}$$

6.3.2 Term-Based Methods

There are many classic term-based approaches. The Rocchio algorithm [36], which has been widely adopted in information retrieval, can build text representation of a training set using a Centroid $\vec{c}$ as follows:

$$\vec{c} = \alpha \frac{1}{|D^+|} \sum_{\vec{d} \in D^+} \frac{\vec{d}}{\|\vec{d}\|} - \beta \frac{1}{|D^-|} \sum_{\vec{d} \in D^-} \frac{\vec{d}}{\|\vec{d}\|} \tag{11}$$

where α and β are empirical parameters; D^+ and D^- are the sets of positive and negative documents, respectively; $\vec{d}$ denotes a document.

Probabilistic methods (Prob) are well-known term-based approaches. The following is the best one:

$$W_{pr}(t) = \log \frac{\frac{(r+0.5)}{(n-r+0.5)}}{\frac{(R-r+0.5)}{(N-n-R+r+0.5)}} \tag{12}$$

where N and R are the total number of documents and the number of positive documents in the training set, respectively; n is the number of documents which contain t; and r is the number of positive documents which contain t.

In addition, TFIDF is also widely used. The term t can be weighted by $W_{tfidf}(t) = TF(d,t) \times IDF(t)$, where term frequency $TF(d,t)$ is the number of times that term t occurs in document $d (d \in D)$ (D is a set of documents in the dataset); $DF(t)$ is the document frequency which is the number of documents that contain term t; and $IDF(t)$ is the inverse document frequency.

Another well-known term-based model is the BM25 approach, which is basically considered the state-of-the-art baseline in IR [35]. The weight of a term t can be estimated by using the following function:

$$W_{bm25}(t) = \frac{TF \cdot (k_1 + 1)}{k_1 \cdot ((1-b) + b\frac{DL}{AVDL}) + TF} \cdot W_{pr}(t) \tag{13}$$

where TF is the term frequency; k_1 and b are the parameters; DL and $AVDL$ are the document length and average document length. The values of k_1 and b are set as 1.2 and 0.75, respectively, according to the suggestion in [47], [48].

The support vector machine (SVM) model is also a well-known learning method introduced by Cortes and Vapnik [8]. Since the works of Joachims [15], [16], researchers have successfully applied SVM to many related tasks and presented some convincing results [5], [6], [27], [39], [55]. The decision function in SVM is defined as:

$$sign(W \cdot x + b) = \begin{cases} +1 & \text{if} \quad (W \cdot x + b) > 0 \\ -1 & \text{else} \end{cases} \tag{14}$$

where x is the input space; $b \in \mathcal{R}$ is a threshold and

$$W = \sum_{i=1}^{l} y_i \alpha_i x_i$$

for the given training data:

$$(x_i, y_i), \ldots, (x_l, y_l) \tag{15}$$

where $x_i \in \mathcal{R}^n$ and y_i equals $+1$ (-1), if document x_i is labeled positive (negative). $\alpha_i \in \mathcal{R}$ is the weight of the training example x_i and satisfies the following constraints:

$$\forall i : \alpha_i \geq 0 \quad \text{and} \quad \sum_{i=1}^{l} \alpha_i y_i = 0. \tag{16}$$

Since all positive documents are treated equally before the process of document evaluation, the value of α_i is set as 1.0 for all of the positive documents and thus the α_i value for the negative documents can be determined by using Eq. (13).

In document evaluation, once the concept for a topic is obtained, the similarity between a test document and the concept is estimated using *inner product*. The relevance of a document d to a topic can be calculated by the function $R(d) = \vec{d} \cdot \vec{c}$, where $\vec{d}$ is the term vector of d and $\vec{c}$ is the concept of the topic.

For both term-based models and CBM, we use the following equation to assign weights for all incoming documents d based on their corresponding W functions:

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

6.4 Hypotheses

The major objective of the experiments is to show how the proposed approach can help improving the effectiveness of pattern-based approaches. Hence, to give a comprehensive investigation for the proposed model, our experiments involve comparing the performance of different pattern-based models, concept-based models and term-based models.

In the experiments, the proposed model is evaluated in term of the following hypothesis:

- Hypothesis H1: The proposed model, PTM(IPE), is designed to achieve the high performance for determining relevant information to answer what users want. The model would be better than other pattern-based models, concept-based models and state-of-the-art term-based models in the effectiveness.

- Hypothesis H2: The proposed deploying method has better performance for the interpretation of discovered patterns in text documents. This deploying approach is not only promising for pattern-based approaches, but also significant for the concept-based model.

In order to compare the proposed approach with others, the baseline models are grouped into three categories as mentioned the above. The first category contains all data mining-based (DM) methods, such as sequential pattern mining, sequential closed pattern mining, frequent itemset mining, frequent closed itemset mining, where $min\_sup = 0.2$. The second category incudes the concept-based model (CBM) that uses the deploying method and the CBM Pattern Matching model; and the last category includes nGram, Rocchio, Probabilistic model, TFIDF, and two state-of-the-art models, BM25 and SVM. A brief of these methods is depicted in Table 4.

TABLE 4
The list of methods used for evaluation.

Method	Description	Algorithm
Sequential ptns.	Data mining method using freq. sequential patterns	SPM
Sequential closed ptns.	Data mining using freq. sequential closed patterns	SCPM
Freq. itemsets	Data mining method using freq. itemsets	NSPM
Freq. closed itemsets	Data mining method using freq. closed itemsets	NSCPM
CBM	Concept with Deploying	Eq. (9)
CBM Ptn matching	Concept with ptn matching	Eq. (10)
nGram	nGram method with $n = 3$	3Gram
Rocchio	Rocchio method	Eq. (11) $\alpha = 1, \beta = 0$
Prob	Probabilistic method	Eq. (12)
TFIDF	TFIDF method	TFIDF Section 6.3.2
BM25	Probabilistic method	Eq. (13) $k_1 = 1.2, b = 0.75$
SVM	Support Vector Machines method	Eq. (14) $b = 0$

6.5 Experimental Results

This section presents the results for the evaluation of the proposed approach PTM (IPE), inner pattern evolving in the pattern taxonomy model. The results of overall comparisons are presented in Table 5, and the summary result are described in Figure 2. We list the result obtained based only on the first 50 TREC topics in Table 5 since not all methods can complete all tasks in the last 50 TREC topics. As aforementioned, itemset-based data mining methods struggle in some topics as too many candidates are generated to be processed. In addition, results obtained based on the first 50 TREC topics are more practical and reliable since the judgment for these topics is manually made by domain experts, whereas the judgment for the last 50 TREC topics is created based on the metadata tagged in each document.

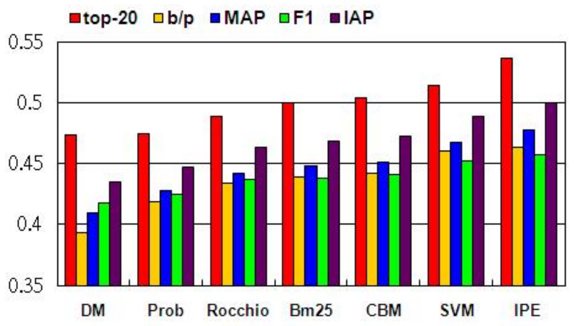

Fig. 2. Comparison of PTM (IPE) and other major models in 5 measures for the 100 topics.

TABLE 5
Comparison of all methods on the first 50 topics.

Method	*top-20*	*b/p*	*MAP*	$F_{\beta=1}$	*IAP*
PTM (IPE)	**0.493**	**0.429**	**0.441**	**0.440**	**0.466**
Sequential ptns	0.401	0.343	0.361	0.385	0.384
Sequential closed ptns	0.406	0.353	0.364	0.390	0.392
Freq. itemsets	0.412	0.352	0.361	0.386	0.384
Freq. closed itemsets	0.428	0.346	0.361	0.385	0.387
CBM	0.448	0.409	0.415	0.423	0.440
CBM Pattern Matching	0.329	0.282	0.283	0.320	0.311
nGram	0.401	0.342	0.361	0.386	0.384
Rocchio	0.416	0.392	0.391	0.408	0.418
Prob	0.407	0.381	0.379	0.396	0.402
TFIDF	0.321	0.321	0.322	0.355	0.348
BM25	0.434	0.399	0.401	0.410	0.422
SVM	0.447	0.409	0.408	0.421	0.434

The most important information revealed in this table is that our proposed PTM (IPE) outperforms not only the pattern mining-based methods, but also the term-based methods including the state-of-the-art methods BM25 and SVM. PTM (IPE) is also outperforms CBM Pattern Matching and CBM in the five measures. CBM outperforms all other models for the first 50 topics. For the time complexity in the testing phase, all models take $O(|T| \times |d|)$ for all incoming documents d. In our experiments, all models used 702 terms for each topic in average. Therefore, there is no significant difference between these models on time complexity in the testing phase.

TABLE 6
Performance of inner pattern evolving in PTM on all topics.

	PDM ($min\_sup = 0.2$)	IPE ($\mu = 5$)
top-20	0.5265	**0.5360**
b/p	0.4598	**0.4632**
MAP	0.4734	**0.4770**
$F_{\beta=1}$	0.4528	**0.4570**
IAP	0.4932	**0.4994**

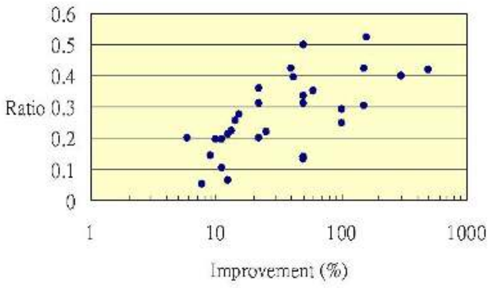

Fig. 3. The relationship between the proportion in number of negative documents greater than threshold to all documents and corresponding improvement on IPE with $\mu = 5$ on improved topics.

6.6 Discussion

6.6.1 PDM to IPE

Table 6 depicts the figures of evaluating measures achieved by inner pattern evolving methods (IPE) and pure pattern deploying method (PDM) on all RCV1 topics. As we can see from the table the evolving method (IPE) outperforms PDM in all measures.

In order to evaluate the effectiveness of PTM (IPE), we attempt to find the correlation between the achieved improvement and the parameter, denoting the ratio of the number of negative documents greater than the threshold to the number of all documents. This value can be obtained using the following equation:

$$Ratio = \frac{|\{d|d \in D^-, weight(d) \geq threshold(DP)\}|}{|D^+| + |D^-|} \qquad (17)$$

Figure 3 illustrates the relationship of the improvement as inner evolving is applied and the above-mentioned value of Ratio. As we can see that the degree of improvement is in direct proportion to the score of Ratio. That means the more qualified negative documents are detected for concept revision, the more improvement we can achieve. In other words, the expected result can be achieved by using the proposed approach.

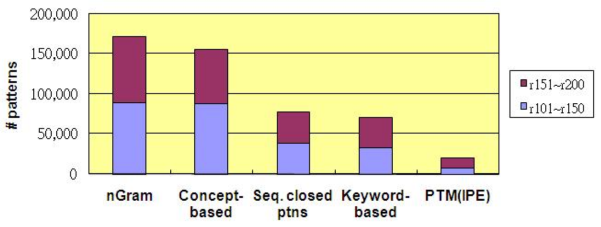

Fig. 4. Comparison in the number of patterns used for training by each method on the first 50 topics (r101∼r150) and the rest of the topics (r151∼r200).

6.6.2 PTM (IPE) vs Other Models

The number of patterns used for training by each method is shown in Figure 4. The total number of patterns is estimated by accumulating the number for each topic. As a result, the figure shows PTM (IPE) is the method that utilizes the least amount of patterns for concept learning compared to others. This is because the efficient scheme of pattern pruning is applied to the PTM (IPE) method. Nevertheless, the classic methods such as Rocchio, Prob and TFIDF adopt terms as patterns in the feature space, they use much more patterns than the proposed PTM (IPE) method and slightly less than the sequential closed pattern mining method. Particularly, nGram and the concept-based models are the methods with the lowest performance which requires more than 15,000 patterns for concept learning. In addition, the total number of patterns obtained based on the first 50 topics is almost the same as the number obtained based on the last 50 topics for all methods except PTM (IPE). The figure based on the first topics group (r101∼r150) for PTM (IPE) is less than that based on the other group (r151∼r200). This can be explained in that the high proportion of closed patterns is obtained by using PTM (IPE) based on the first topics group.

A further investigation in the comparison of PTM (IPE) and TFIDF in *top-20* precision on all RCV1 topics is depicted in Figure 5. It is obvious that PTM (IPE) is superior to TFIDF as it can be seen that positive results distribute over all topics, especially for the first 50 topics. Another observation is the scores on the first 50 topics are better than those on the last fifty. That is because of the different ways of generating these two sets of topics, which has been mentioned before. The interesting behavior is that there are a few topics where TFIDF outperforms PTM. After further investigation, we found a similar characteristic of these topics in that there are only a few positive examples available in these topics. For example, topic r157, which is the worst case for PTM (IPE) compared to TFIDF, has only three positive documents available. Note that the average number of positive documents for each topic is over 12. The similar behaviors are found in topic r134 and r144.

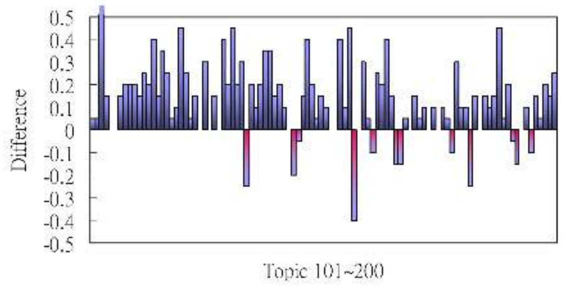

Fig. 5. Comparison of PTM (IPE) and TFIDF in *top-20* precision.

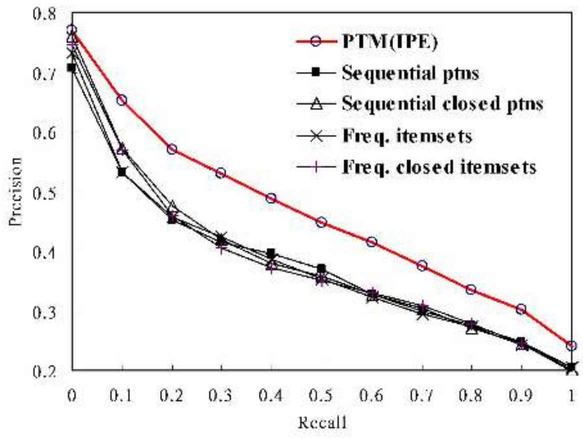

Fig. 6. Comparing PTM (IPE) with Data Mining methods on the first 50 TREC topics.

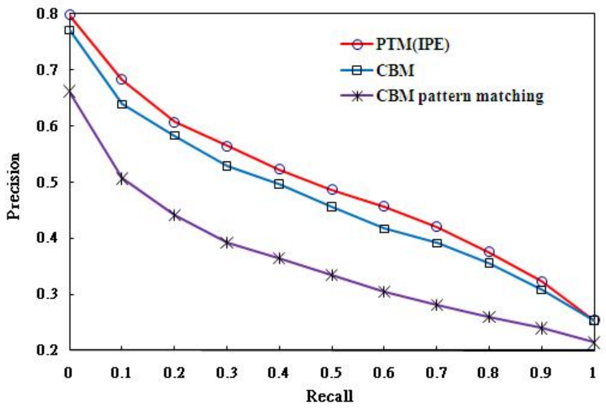

Fig. 7. Comparing PTM (IPE) with concept-based models on the all 100 TREC topics.

The plotting of precisions on 11 standard points for PTM (IPE) and pattern mining based methods on the first 50 topics is illustrated in Figure 6. The result supports the superiority of the PTM (IPE) method and highlights the importance of the adoption of proper pattern deploying and pattern evolving methods to a pattern-based knowledge discovery system. Comparing their performance at the first few points around the low-recall area, it is also found that the points for pattern mining methods drop rapidly as the recall value rises and then keep a relatively gradual slope from the mid recall period to the end. All four pattern mining methods achieve similar results. However, the plotting curve for PTM (IPE) is much smoother than those for pattern mining methods as there is no severe fluctuation on it. Another observation on this figure is that the pattern mining-based methods however perform well at the point where recall is close to zero, despite the overall unpromising results they have. Accordingly, we can conclude that the pattern mining-based methods can improve the performance in the low-recall situation.

Although the PTM (IPE) is equipped with the pattern mining algorithm for discovering sequential closed patterns, the promising results cannot be produced without the help from the successful application of the proposed d-patterns and inner pattern evolving. The proper usage of d-patterns, which has been proven previously, can overcome the misinterpretation problem and provide a feasible solution to effectively exploit the vast amount of patterns generated by data mining algorithms. Moreover, the employment of IPE provides the mechanism to utilize the information from negative examples to overcome the low-frequency problem. In conclusion, the experimental results provide evidences showing that the PTM (IPE) method is an ideal model for further developing pattern mining based approaches.

As mentioned in the last sub-section, PTM (IPE) is outperforms CBM Pattern Matching and CBM in all 5 measures and CBM outperforms all other models for the first 50 topics. It looks that the concept-based model has the promising potential for improving the performance of text mining in the future. Figure 7 shows the plotting of precision on 11 standard points for PTM (IPE), CBM, and CBM Pattern Matching. It also shows that the deploying approach for using concepts to answer what users want is significant for the concept-based model because CBM is much better than the CBM Pattern Matching model. In general, the PTM (IPE) method outperforms CBM in these experiments.

Figure 8 presents the plotting of precisions at 11 standard points for PTM (IPE) and term-based methods on the first 50 topics. Compared to the previous plotting in Figure 6, the difference of performance for all methods is easier to be recognized in the figure. Again, the PTM (IPE) method outperforms all other methods. Among these methods, the nGram method achieves a noticeable score of precision at the first point where recall equals to zero, meaning that the nGram method is able to promote top relevant documents toward the front of the ranking list. As mentioned before, data mining-based methods can

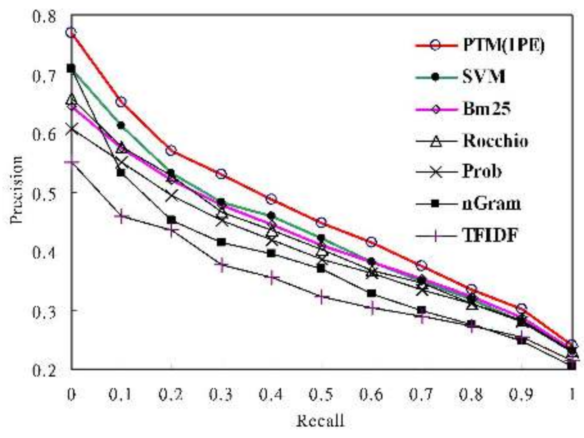

Fig. 8. Comparing PTM (IPE) with Term-based methods on the first 50 TREC topics.

perform well at the low-recall area, which can explain why nGram has better results at this point. However, the scores for the nGram method drop rapidly at the following couple of points. During that period, SVM, BM25, Rocchio and Prob methods transcend the nGram method and keep the superiority until the last point where recall equals to 1. There is no doubt that the lowest performance is produced by the TFIDF method, which outperforms the nGram method only at the last few recall points. In addition, the Prob method is superior to the nGram method, but inferior to the Rocchio method. The overall performance of Rocchio is better than that for the Prob method which corresponds to the finding in [50].

In summary, the proposed approach PTM (IPE) achieves an outstanding performance for text mining by comparing with the up-to-date data mining-based methods, the concept models, and the well-known term-based methods, including the state-of-the-art BM25 and SVM models. The results show the PTM (IPE) model can produce encouraging gains in effectiveness, in particular over the SVM and CBM models. These results strongly support Hypothesis H1. The promising results can be explained in that the use of the deploying method is promising (Hypothesis H2 is also supported) for solving the misinterpretation problem because it can combine well with the advantages of terms and discovered patterns or concepts. Moreover, the inner pattern deploying strategy provides an effective evaluation for reducing the side-effects of noisy patterns because the estimation of term weights in the term space is based on not only terms' statistical properties but also patterns' associations in the corresponding pattern taxonomies.

7 Conclusions

Many data mining techniques have been proposed in the last decade. These techniques include association rule mining, frequent itemset mining, sequential pattern mining, maximum pattern mining and closed pattern mining. However, using these discovered knowledge (or patterns) in the field of text mining is difficult and ineffective. The reason is that some useful long patterns with high specificity lack in support (i.e., the low-frequency problem). We argue that not all frequent short patterns are useful. Hence, misinterpretations of patterns derived from data mining techniques lead to the ineffective performance. In this research work, an effective pattern discovery technique has been proposed to overcome the low-frequency and misinterpretation problems for text mining. The proposed technique uses two processes, pattern deploying and pattern evolving, to refine the discovered patterns in text documents. The experimental results show that the proposed model outperforms not only other pure data mining-based methods and the concept-based model, but also term-based state-of-the-art models, such as BM25 and SVM based models.

8 Acknowledgments

This paper was partially supported by Beijing Natural Science Foundation (4102007) and Grant DP0988007 from the Australian Research Council (ARC Discovery Project). The authors also wish to thank Prof. Peter Bruza, Dr. Raymond Lau, Dr. Yue Xu, Dr. Xiaohui Tao, and Dr. Xujuan Zhou for their suggestions. In addition, we would like to thank anonymous reviewers for their constructive comments.

References

[1] K. Aas and L. Eikvil. Text categorisation: A survey. Technical report, Norwegian Computing Center, Raport NR 941, 1999.

[2] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, pages 478–499, 1994.

[3] H. Ahonen, O. Heinonen, M. Klemettinen, and A. I. Verkamo. Applying data mining techniques for descriptive phrase extraction in digital document collections. In *Proceedings of the IEEE Forum on Research and Technology Advances in Digital Libraries (ADL98)*, pages 2–11, 1998.

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

[5] N. Cancedda, N. Cesa-Bianchi, A. Conconi, and C. Gentile. Kernel methods for document filtering. In *TREC (URL: trec.nist.gov/ pubs/ trec11/ papers/ kermit.ps.gz)*, 2002.

[6] N. Cancedda, E. Gaussier, C. Goutte, and J-M. Renders. Word-sequence kernels. *Journal of Machine Learning Research*, 3:1059–1082, 2003.

[7] M. F. Caropreso, S. Matwin, and F. Sebastiani. Statistical phrases in automated text categorization. Technical report, Instituto di Elaborazione dell'Informazione, Technical Report IEI-B4-07-2000, 2000.

[8] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[9] S. T. Dumais. Improving the retrieval of information from external sources. *Behavior Research Methods, Instruments, & Computers*, 23(2):229–236, 1991.

[10] J. Han and K.C-C. Chang. Data mining for web intelligence. *Computer*, 35(11):64–70, 2002.

[11] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *Proceedings of ACM-SIGMOD 2000*, pages 1–12, 2000.

[12] Y. Huang and S. Lin. Mining sequential patterns using graph search techniques. In *Proceedings of the 27th Annual International Computer Software and Applications Conference*, pages 4–9, 2003.

[13] N. Jindal and B. Liu. Identifying comparative sentences in text documents. In *Proceedings of SIGIR 2006*, pages 244–251, 2006.

[14] T. Joachims. A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. In *Proceedings of the 14th International Conference on Machine Learning (ICML 1997)*, pages 143–151, 1997.

[15] T. Joachims. Text categorization with suport vector machines: Learning with many relevant features. In *Proceedings of the European Conference on Machine Learning*, pages 137–142, 1998.

[16] T. Joachims. Transductive inference for text classification using support vector machines. In *Proceedings of the 16th International Conference on Machine Learning (ICML 1999)*, pages 200–209, 1999.

[17] W. Lam, M. E. Ruiz, and P. Srinivasan. Automatic text categorization and its application to text retrieval. *IEEE Transactions on Knowledge and Data Engineering*, 11(6):865–879, 1999.

[18] D. D. Lewis. An evaluation of phrasal and clustered representations on a text categorization task. In *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1992)*, pages 37–50, 1992.

[19] D. D. Lewis. Feature selection and feature extraction for text categorization. In *Speech and Natural Language Workshop*, pages 212–217, 1992.

[20] D. D. Lewis. Evaluating and optimizing automous text classification systems. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 246–254, 1995.

[21] X. Li and B. Liu. Learning to classify texts using positive and unlabeled data. In *Proceedings of International Joint Conferences on Artificial Intelligence (IJCAI 2003)*, pages 587–594, 2003.

[22] Y. Li, W. Yang, and Y. Xu. Multi-tier granule mining for representations of multidimensional association rules. In *Proceedings of 6th IEEE International Conference on Data Mining (ICDM 2006), Hong Kong*, pages 953–958, 2006.

[23] Y. Li, C. Zhang, and J. R. Swan. An information filtering model on the web and its application in jobagent. *Knowledge-based Systems*, 13(5):285–296, 2000.

[24] Y. Li and N. Zhong. Interpretations of association rules by granular computing. In *Proceedings of 3rd IEEE International Conference on Data Mining (ICDM03), USA*, pages 593–596, 2003.

[25] 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.

[26] Y. Li, X. Zhou, P. Bruza, Y. Xu, and R. Y. Lau. A two-stage text mining model for information filtering. In *proceedings of ACM 17th Conference on Information and Knowledge Management (CIKM 2008), Napa Valley, California, USA*, pages 1023–1032, 2008.

[27] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. Text classification using string kernels. *Journal of Machine Learning Research*, 2:419–444, 2002.

[28] A. Maedche. *Ontology learning for the semantic Web*. Kluwer Academic, 2003.

[29] C. Manning and H. Schütze. *Foundations of statistical natural language processing*. MIT Press, Cambridge, MA, USA, 1999.

[30] I. Moulinier, G. Raskinis, and J. Ganascia. Text categorization: A symbolic approach. In *Proceedings of the 5th Annual Symposium on Document Analysis and Information Retrieval (SDAIR)*, pages 87–99, 1996.

[31] J. S. Park, M. S. Chen, and P. S. Yu. An effective hash-based algorithm for mining association rules. In *Proceedings of ACM SIGMOD Conference*, pages 175–186, 1995.

[32] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M. Hsu. Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth. In *Proceedings of the 17th International Conference on Data Engineering (ICDE 2001)*, pages 215–224, 2001.

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

[34] S. Robertson and I. Soboroff. The trec 2002 filtering track report. In *TREC 2002 (URL:trec.nist.gov/ pubs/ trec11/ papers/ OVER.FILTERING.ps.gz)*, 2002.

[35] S. E. Robertson, S. Walker, and M. Hancock-Beaulieu. Experimentation as a way of life: Okapi at trec. *Information Processing and Management*, 36(1):95–108, 2000.

[36] J. Rocchio. *Relevance Feedback in Information Retrieval*, chapter 14, pages 313–323. Prentice-Hall, 1971.

[37] T. Rose, M. Stevenson, and M. Whitehead. The reuters corpus volume1 - from yesterday's news to today's language resources. In *Proceedings of the 3rd Inter. Conf. on Language Resources and Evaluation*, pages 29–31, 2002.

[38] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management: an International Journal*, 24(5):513–523, 1988.

[39] M. Sassano. Virtual examples for text classification with support vector machines. In *Proceedings of Empirical Methods in Natural Language Processing*, pages 208–215, 2003.

[40] S. Scott and S. Matwin. Feature engineering for text classification. In *Proceedings of the Sixteenth International Conference on Machine Learning (ICML 1999)*, pages 379–388, 1999.

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

[42] M. Seno and G. Karypis. Slpminer: An algorithm for finding frequent sequential patterns using length-decreasing support constraint. In *Proceedings of the 2nd IEEE International Conference on Data Mining (ICDM 2002)*, pages 418–425, 2002.

[43] R. E. Shapire and Y. Singer. Boostexter: a boosting-based system for text categorization. *Machine Learning*, 39:135–168, 2000.

[44] R. Sharma and S. Raman. Phrase-based text representation for managing the web document. In *Proceedings of The International Conference on Information Technology: Computers and Communications (ITCC)*, pages 165–169, 2003.

[45] Shady Shehata, Fakhri Karray, and Mohamed Kamel. Enhancing text clustering using concept-based mining model. In *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM 2006)*, pages 1043–1048, Hong Kong, 2006.

[46] Shady Shehata, Fakhri Karray, and Mohamed Kamel. A concept-based model for enhancing text categorization. In *Proceedings of The 13th International Conference on Knowledge Discovery and Data Mining (KDD 2007)*, pages 629–637, San Jose, California, USA, 2007.

[47] K. Sparck Jones, S. Walker, and S. E. Robertson. A probabilistic model of information retrieval: development and comparative experiments - part 1. *Information Processing and Management*, 36(6):779–808, 2000.

[48] K. Sparck Jones, S. Walker, and S. E. Robertson. A probabilistic model of information retrieval: development and comparative experiments - part 2. *Information Processing and Management*, 36(6):809–840, 2000.

[49] R. Srikant and R. Agrawal. Mining generalized association rules. In *Proceedings of 21th International Conference on Very Large Data Bases (VLDB 1995)*, pages 407–419, 1995.

[50] S-T. Wu, Y. Li, and Y Xu. Deploying approaches for pattern refinement in text mining. In *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM 2006)*, pages 1157–1161, 2006.

[51] S-T. Wu, Y. Li, Y. Xu, B. Pham, and P. Chen. Automatic pattern-taxonomy extraction for web mining. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence (WI04)*, pages 242–248, 2004.

[52] Y. Xu and Y. Li. Generating concise association rules. In *Proceedings of ACM 16th Conference on Information and Knowledge Management (CIKM 2007)*, pages 781–790, 2007.

[53] X. Yan, J. Han, and R. Afshar. Clospan: mining closed sequential patterns in large datasets. In *Proceedings of SIAM Int. Conf. on Data Mining (SDM03)*, pages 166–177, 2003.

[54] Y. Yang. An evaluation of statistical approaches to text categorization. *Information Retrieval*, 1:69–90, 1999.

[55] Y. Yang and X. Liu. A re-examination of text categorization methods. In *Proceedings of the Annual ACM SIGIR Conference (SIGIR 1999)*, pages 42–49, 1999.

[56] M. Zaki. Spade: An efficient algorithm for mining frequent sequences. *Machine Learning*, 40:31–60, 2001.

Ning Zhong is currently Head of the Knowledge Information Systems Laboratory and is a Professor in the Department of Life Science and Informatics at Maebashi Institute of Technology, Japan. He is also director and an adjunct professor in the International WIC Institute (WICI), Beijing University of Technology. He has conducted research in the areas of knowledge discovery and data mining, rough sets and granular-soft computing, Web intelligence, intelligent agents, brain informatics, and knowledge information systems, with more than 200 journal and conference publications and 20 books. He has been the chair of the IEEE Computer Society Technical Committee on Intelligent Informatics (TCII) in 2006 to 2009. He is currently the Editor-in-Chief of Web Intelligence and Agent Systems and serves as associate editor/editorial board for several international journals and book series. He is also a co-chair of Web Intelligence Consortium (WIC), and the chair of IEEE Computational Intelligence Society Task Force on Brain Informatics.

Yuefeng Li is the Leader of the eDiscovery Lab in the Institute for Creative Industries and Innovation, and a Professor in the Discipline of Computer Science, Faculty of Science and Technology, Queensland University of Technology, Australia. He has published over 120 refereed papers (including 30 journal papers). He is also a co-author of one book, and an editor of five books. He has supervised six PhD students and four Master by research students to successful completion. He is an Associate Editor of the International Journal of Pattern Recognition and Artificial Intelligence and an Associate Editor of the IEEE Intelligent Informatics Bulletin. He has established a strong reputation internationally in the fields of Web Intelligence, Text Mining and Ontology Leaning, and has been awarded three Australian Research Council grants.

Sheng-Tang Wu received the MS and PhD degrees in the Faculty of Information Technology at Queensland University of Technology, Brisbane, Australia in 2003 and 2007, respectively. He also received the honor of the Outstanding Doctoral Thesis Award at Queensland University of Technology. Dr. Wu is currently an Assistant Professor in the Department of Information Science and Applications, Asia University, Taiwan. His research interests include data mining, Web intelligence, information retrieval, information systems and multimedia.