INTERNATIONAL JOURNAL OF RESEARCH IN COMPUTER APPLICATIONS AND ROBOTICS
ISSN 2320-7345

TEXT EXTRACTION USING EFFICIENT PROTOTYPE

[1]K.S.ARCHANA, [2]ASWANI KUMAR UNNAM

[1]PG Scholar, Department of Information Technology, TKR College of Engineering and Technology
Hyderabad, A.P-500 097, India.
[2]Associatei Professor, Department of Information Technology, TKR College of Engineering and Technology2
Hyderabad, A.P-500 097, India.
[1]Arch.hebbar@gmail.com ,[2]asphky@gmail.com

Abstract—
Various data mining techniques have been proposed for mining useful Models in text documents. However, how to effectively use and update discovered Models 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 difficulties of polysemy and synonymy. Over the years, people have often held the hypothesis that Prototype based approaches should perform better than the term-based ones, but various experiments do not support this hypothesis. Here we presents an innovative and effective Prototype discovery technique which includes the processes of Prototype deploying and Prototype evolving, to improve the effectiveness of using and updating discovered Models for finding relevant and interesting information. Substantial experiments on RCV1 data collection and TREC topics demonstrate that the proposed solution achieves encouraging performance.
.

Key words—Text mining, text classification, Prototype mining, Prototype evolving, information filtering

I INTRODUCTION

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. Various 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 item set mining, sequential Prototype mining, maximum Prototype mining, and closed Prototype mining. Most of them are proposed for the purpose of developing efficient mining algorithms to find particular Models within a reasonable and acceptable time frame. With a large number of Models generated by using data mining approaches, how to effectively use and update these

Models 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 Models 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 in text documents to help users to find what they want. In the beginning, Information Retrieval provided various term-based methods to solve this challenge, such as Rocchio and probabilistic models, rough set models, BM25 and support vector machine 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 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 various 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. Although phrases are less ambiguous and more discriminative than individual terms, the likely reasons for the discouraging performance include:

1) Phrases have inferior statistical properties to terms,
2) They have low frequency of occurrence, and
3) There are large numbers of redundant and noisy phrases among them.

In the presence of these setbacks, sequential Models used in data mining community have turned out to be a promising alternative to phrases, because sequential Models enjoy good statistical properties like terms. To overcome the disadvantages of phrase-based approaches, Prototype mining-based approaches have been proposed, which adopted the concept of closed sequential Models, and pruned no closed Models. These Prototype mining-based approaches have shown certain extent improvements on the effectiveness. However, the paradox is that people think Prototype-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 Prototype-based approaches: low frequency and misinterpretation. Given a specified topic, a highly frequent Prototype is usually a general Prototype, or a specific Prototype of low frequency. If we decrease the minimum support, a lot of noisy Models would be discovered. Misinterpretation means the measures used in Prototype mining turn out to be not suitable in using discovered Models to answer what users want. The difficult problem hence is how to use discovered Models to accurately evaluate the weights of useful features in text documents.

IR has developed various mature techniques which demonstrated that terms were important features in text documents. However, various terms with larger weights (e.g., the term frequency and inverse document frequency 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 Prototype discovery technique, which first calculates discovered specificities of Models and then evaluates term weights according to the distribution of terms in the discovered Models rather than the distribution in documents for solving the misinterpretation problem. It also considers the influence of Models from the negative training examples to find ambiguous (noisy) Models and try to reduce their influence for the low-frequency problem. The process of updating ambiguous Models can be referred as Prototype evolution. The proposed approach can improve the accuracy of evaluating term weights because discovered Models are more specific than whole documents.

II. ASSOCIATED EXERTION

Various types of text representations have been proposed in the past. A well known one is the bag of words that uses keywords as elements in the vector of the feature space. In, 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 and improves performance by an average of 30 percent. Various weighting schemes for the bag of words representation approach were given in. 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 over fitting.

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. With respect to the representation of the content of documents, some research works have used phrases rather than individual words. In , the combination of unigram and bigrams was chosen for document indexing in text categorization and evaluated on a variety of feature evaluation functions (FEF). A phrase-based text representation for Web document management was also proposed in .In 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. Term-based ontology mining methods also provided some thoughts for text representations. For example, hierarchical clustering was used to determine synonym-my and hyponymy relations between keywords. Also, the Prototype evolution technique was introduced in in order to improve the performance of term-based ontology mining. Prototype mining has been extensively studied in data mining communities for various years. A variety of efficient algorithms such as Apriority-like algorithms Prefix Span FP-tree SPADE, SLP Miner, and GST have been proposed. These research works have mainly focused on developing efficient mining algorithms for discovering Models from a large data collection. However, searching for useful and interesting Models and rules was still an open problem. In the field of text mining, Prototype mining techniques can be used to find various text Models, such as sequential Models, frequent item sets, 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 Models. For the challenging issue, closed sequential Models have been used for text mining in, which proposed that the concept of closed Models in text mining was useful and had the potential for improving the performance of text mining.

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	$t_1\ t_2\ t_6\ t_7$

In addition a two-stage model that used both term-based methods and Prototype-based methods was introduced in to significantly improve the performance of information filtering. Natural language processing 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 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 to describe the semantic 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 no 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.

III Prototype Taxonomy Model

If all the 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 $T = (t_1, t_2, \ldots, t_m)$ be a set of terms which can be extracted from the set of positive documents, D^+.

A. Frequent and Closed Models

Given a term set X in document d, ˊXˋ is used to denote the covering set of X for d, which includes all paragraphs dp ϵ
PS (d) such that X ≤ dp, i.e., ˊXˋ ={dp|dp ϵ PS(d),X ≤ dp}.

Its absolute support is the number of occurrences of X in PS(d), that is sup (X) = | ˊXˋ |. Its relative support is the Fraction of the paragraphs that contain the Prototype, that is,supr(X) = |ˊXˋ| /| PS(d)|. A term set X is called frequent Prototype if its sup r (or sup a)≥ min sup, a minimum support.

Table 1 lists a set of paragraphs for a given document d, where PS(d) = {dp1, dp2, . . . , dp6}, and duplicate terms were removed

TABLE 2
Frequent Models and Covering Sets

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

. Let min_sup = 50%, we can obtain ten frequent Models in Table 1 using the above definitions. Table 2 illustrates the ten frequent Models and their covering sets.

Not all frequent Models in Table 2 are useful. For example, Prototype {t_3, t_4 } always occurs with term t_6 in paragraphs, i.e., the shorter Prototype, {t_3, t_4,} is always a part of the larger Prototype, {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 Prototype and expect to keep the larger Prototype, {t3, t4, t6 }, only.

Given a term set X, its covering set ˊXˋ is a subset of paragraphs. Similarly, given a set of paragraphs Y ≤ PS(d), we can define its term set, which satisfies

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

The closure of X is defined as follows: C ls (X) = term set (ˊXˋ)

A Prototype X (also a termset) is called closed if and only if X = Cls(X).Let X be a closed Prototype. We can prove that

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

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

$$\lceil X_1 \rceil = \lceil X \rceil,$$

where $sup_a(X_1)$ and $sup_a(X)$ are the absolute support of Prototype X_1 and X, respectively. We also have

$$Cls(X) = termset(\lceil X \rceil) = termset(\lceil X_1 \rceil) \supseteq X_1 \supset X,$$ that is, Cls(X) ≠ X.

B. Prototype Taxonomy

Models 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 Models in Table 2 if assuming min_sup = 50%. There Are, however, only three

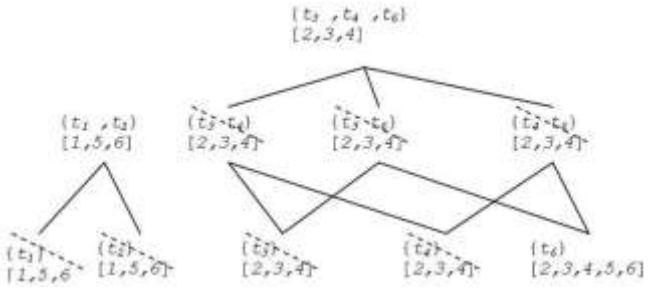

Fig. 1. Prototype taxonomy

Closed Models in this example. They are $<t_3, t_4, t_6>$,$<t_1, t_2>$, and $<t_6>$.

Fig. 1 illustrates an example of the Prototype taxonomy for the frequent Models in Table 2, where the nodes represent frequent Models and their covering sets, non closed Models can be pruned, the edges are "is-a" relation. After pruning, some direct "is-a" retaliations may be changed, for example, Prototype ft_6g would become a direct sub Prototype of $\{t_3, t_4, t_6\}$ after pruning unenclosed Models. Smaller Models in the taxonomy, for example Prototype $\{t_6\}$, (see Fig. 1) are usually more general because they could be used frequently in both positive and negative documents, and larger Models, for example Prototype $\{t_3, t_4, t_6\}$, in the taxonomy are usually more specific since they may be used only in positive documents. The semantic information will be used in the Prototype taxonomy to improve the performance of using closed Models in text mining, which will be further discussed in the next section.

C. Closed Sequential Models

A sequential Prototype $s = <t_1, \ldots, t_r>$ ($t_i \in T$) is an ordered list of terms. A sequence $s_1 = <x_1, \ldots, x_i>$ is a subsequence 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 $1 \_ j_1 < j_2 \ldots < j_y \_ j$ and $x_1 = y_{j1}$, $x_2 = y_{j2}$, . . . , $x_i = y_{jy}$. Given $s_1 \sqsubseteq s_2$, we usually say s_1 is a sub Prototype of s_2, and s_2 is a super Prototype of s_1. In the following, we simply say Models for sequential Models. is still used to denote the covering set of X, which includes A sequential Prototype X is called frequent Prototype if its relative support (or absolute support) _ min sup, a mini-mum support. The property of closed Models (see eq. (1)) can be used to define closed sequential Models. A frequent sequential Prototype X is called closed if not 9 any super-Prototype X_1 of X such that $sup_a(X_1) = sup(X)$.

IV. Prototype Deploying Method

In order to use the semantic information in the Prototype taxonomy to improve the performance of closed Models in text mining, we need to interpret discovered Models by summarizing them as d-Models in order to accurately evaluate term weights. The rational behind this motivation is that d-Models. The evaluation of term weights 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 Models.

A. Representations of Closed Models

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

$$p_1 \oplus p_2 = \{(t, x_1\ x_2) | (t, x_1) \in p_1, (t, x_{\bar{2}}) \in p_2\} \cup \{(t, x) | (t, x) \in p_1 \cup p_2, \text{not}((t, \_) \in p_1 \cap p_2)\},$$

where is the wild card that matches any number. For the special case we have $p \oplus \phi = p$; 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 Models on a common set of terms T in order to obtain the following d-Models (deployed Models, non sequential weighted Models): $\hat{d}_i = \{(t_{i1}, n_{i1}), (t_{i2}, n_{i2}), \ldots, (t_{im}, n_{im})\}$ where t_{ij} in pair (t_{ij}, n_{ij}) denotes a single term and n_{ij} is its support in d_i which is the total absolute supports given by closed Models that contain t_{ij}, or n_{ij} (simply in this paper) is the total number of closed Models that contain t_{ij}.

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

$\hat{d}_b = \{(t_1, 3), (t_2, 3), (t_3, 3), (t_4, 3), (t_6, 8)\}$;

The process of calculating d-Models can be easily described by using the $\oplus$ operation in Algorithm 1 (PTM) shown in Fig. 2 that will be described in the next section, where a term's support is the total number of closed Models that contain the term.

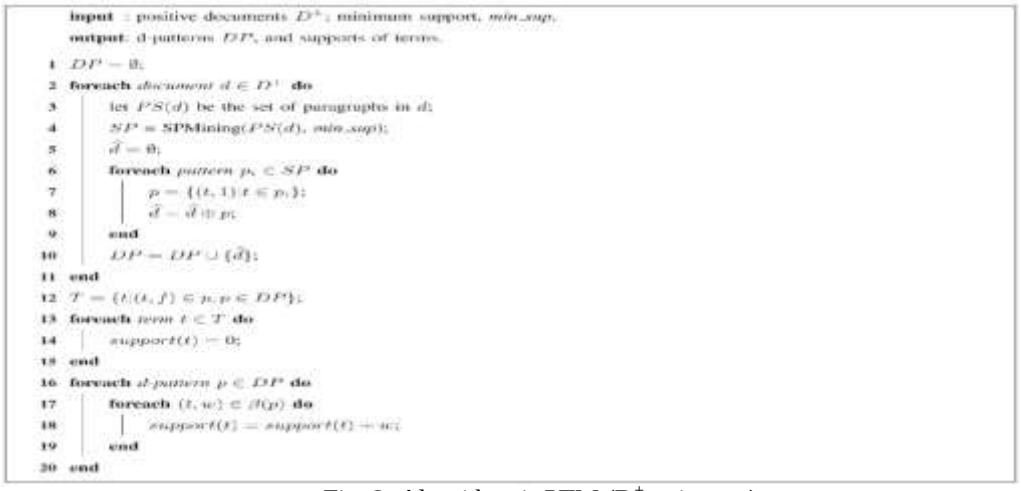

Fig. 2. Algorithm 1: PTM (D^+, min_sup).

$\hat{d}_1 = \{(carbon, 2), (emiss, 1), (air, 1), (pollut, 1)\}$,

$\hat{d}_2 = \{(greenhous, 1), (global, 2), (emiss, 1)\}$,

$\hat{d}_3 = \{(greenhous, 1), (global, 1), (emiss, 1)\}$,

$\hat{d}_4 = \{(carbon, 1), (air, 2), (antarct, 1)\}$,

$\hat{d}_5 = \{(emiss, 1), (global, 1), (pollut, 1)\}$;

Let DP be a set of d-Models in D^+, and $p \in DP$ be a d-Prototype. We call p(t) the absolute support of term t, which is the number of Models that contain t in the corresponding Models taxonomies. In order to effectively deploy Models in different taxonomies from the different positive documents, d-Models 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-Models 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 $(p_i) = \{(t_1, w_1), (t_2, w_2), \ldots, (t_k, w_k)\}$,

for all $p_i \in DP$, where

$$p_i = \{(t_1, f_1), (t_2, f_2), \ldots, (t_k, f_k)\} \in DP, w_i = \frac{f_i}{\sum_{j=1}^{k} f_j}$$

and $T = \{t|(t, f) \in p, p \in DP\}$.

$\beta(p_i)$ is called the normal form (or normalized d-Prototype) of d-Prototype p_i in this paper, and $termset(p_i) = \{t_1, t_2, \ldots, t_k\}$.

C.D-Prototype Mining Algorithm

To improve the efficiency of the Prototype taxonomy mining, an algorithm, SP Mining, was proposed in to find all closed sequential Models, which used the well-known A priori property in order to reduce the searching space. Algorithm 1 (PTM) shown in Fig. 2 describes the training process of finding the set of d-Models. For every positive document, the SP Mining algorithm is first called in step 4 giving rise to a set of closed sequential Models SP . The main focus of this paper is the deploying process, which consists of the d-Prototype discovery and term support evaluation. In Algorithm 1 (Fig. 2), all discovered Models in a positive document are composed into a d-Prototype giving rise to a set of d-Models DP in steps 6 to 9. Thereafter, from steps 12 to 19, term supports are

Example of a Set of Positive Documents Consisting of Prototype Taxonomies
TABLE 3

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	$PT_{(5,1)}$	$\{\langle emiss, global, pollut\rangle_2\}$

The number beside each sequential Prototype indicates the absolute support of Prototype. Calculated based on the normal forms for all terms in d-Models.

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 Models in a positive document, and k be the average number of terms in a discovered Prototype. We also assume that the basic operation is a comparison between two terms.

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

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

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

$$weight(d) = \sum_{t \epsilon T} support(t) \mathbb{T}(t, d), \quad (4)$$

5 Inner Prototype Evolution

How to reshuffle supports of terms within normal forms of d-Models based on negative documents in the training set. The technique will be useful to reduce the side effects of noisy Models because of the low-frequency problem. This technique is called inner Prototype evolution here, because it only changes a Prototype's term supports within the Prototype.

A threshold is usually used to classify documents into relevant or irrelevant categories. Using the d-Models, the threshold can be defined naturally as follows:

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

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-Models have been used to give rise to such an error. We call these Models offenders of nd. An offender of nd is a d-Prototype that has at least one term in nd. The set of offenders of nd is defined by:

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

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 Models is explained as follows: complete conflict offenders are removed from d-Models first. For partial conflict offenders, their term supports are reshuffled in order to reduce the effects of noise documents.

The main process of inner Prototype evolution is implemented by the algorithm IP Evolving . The inputs of this algorithm are a set of d-Models DP , a training set $D = D^+ \cup D^-$. The output is a composed of d-Prototype. Step 2 in IP Evolving is used to estimate the threshold for finding the noise negative documents. Steps 3 to 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-Models NDP. Step 6 calls algorithm Shuffling to update NDP according to noise documents. Steps 7 to 9 compose updated normal forms together. The time complexity of Algorithm 2 in Fig. 3 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 Prototype nd, the algorithm gets its offenders that takes O(nm jndj) 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-Prototype. 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-Models) are removed since all elements within the d-Models are held by the negative documents indicating that they can be discarded for preventing interference from these possible "noises."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-Prototype

where these terms also appear in a noise document. The algorithm calculates the base in step 5 which is certainly not zero since term set(p) nd ≠ 0; and then updates the support distributions of terms in step 6.For example, for the following d-Prototype

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

input : a training set $D = D^+ \cup D^-$; a set of d-patterns DP; and an experimental coefficient μ.

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, μ. NDP); //call Alg. 3
7 **foreach** $p \in NDP$ **do**
8 $np \leftarrow np \oplus p$;
9 **end**
10 **end**

Fig. 3. Algorithm 2: IPEvolving ($D^{)}$, D^{-}, DP , _).

input : a noise document nd, its offenders $\Delta(nd)$, normal forms of d-patterns NDP, and an experimental coefficient μ.

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 $offering = (1 - \frac{1}{\mu}) \times \sum_{t \in (termset(p) \cap nd)} support(t)$;
5 $base = \sum_{t \in (termset(p) - nd)} support(t)$;
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**

Fig. 4. Algorithm 3: Shuffling (nd, _(nd), NDP , _. NDP).

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_9g, $d^{\wedge}$ will be a partial conflict offender since

termset($d^{\wedge}$) ∩ nd = {t_1, t_2, t_6 }=0 ,.

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 ∩ T , where T = {t ϵ term set(p)|p ϵ DP }. Hence, the time complexity of algorithm Shuffling is $O(nm^2)$ for a given noise negative document. Based on the above analysis about Algorithms 2 and 3, the total time complexity of the inner Prototype evolution is O(nm) q(nmjndj) nm^2)) nm^2) = $O(qnm^2)$ considering that the noise negative document nd can be replaced by nd ∩ T before conducting the Prototype 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-Models in positive documents (D^+) based on a min sup, and evaluates term supports by deploying d-Models 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 _. 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.

VI . Evaluation And Discussion

A. Experimental Data Set

The most popular used data set currently is 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 for RCV1 . 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, stop word removal is applied and the Porter algorithm [33] is selected for suffix stripping. Terms with term frequency equalling to one are discarded.

B .Measures

Several standard measures based on precision and recall is 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 experiment evaluation is F_β-measure [20], which combines precision

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

where _ 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 = 1 is adopted in our experiments meaning that it attributes equal importance to precision and recall. When = 1, the measure is expressed as:

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

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 3. In the following, we introduce other two classes: the concept-based model and term-based methods.

C. Concept-Based

A new concept-based model was presented in and 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 labelled 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:

$$weight(c) = tf_{weight}(c) \text{) } ctf_{weight}(c):$$

To have a uniform representation, in this paper, we call a concept as a concept-Prototype which is a set of terms. For example, verb "hits" is denoted as fhitsg and its argument "the ball" is denoted as the, ballg.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 4,000 features at most, the concept-based model uses all features for each topic. Let CP_i be the set of concepts in d_i 2 $D^{)}$. To synthesize both tf and ctf of concepts in all positive documents, we use the following equation to evaluate term weights

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

for all t ϵ T .We also designed another kind of the concept-based model, called CBM Prototype Matching, which evaluates a document d's relevance by accumulating the weights of concepts that appear in d as follows:

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

Term-Based Methods: There are various 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 ~c as follows:

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

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

$$W(t) = \log\left(\frac{r + 0.5}{R - r + 0.5} \Big/ \frac{n - r + 0.5}{(N - n) - (R - r) + 0.5)} \right), \quad (12)$$

where N and R are the total number of documents and the number of positive documents in the training set, respec-tively,

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(t) = TF (d, t) X DF (t), where term frequency TF (d, t) is the number of times that term t occurs in document d(d 2 D) (D is a set of documents in the data set), 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. The weight of a term t can be estimated by using the following function.

$$W(t) = \frac{TF \cdot (k_1 + 1)}{k_1 \cdot ((1-b) + b\frac{DL}{AVDL}) + TF} \cdot \log\frac{(r+0.5)/(n-r+0.5)}{(R-r+0.5)/(N-n-R+r+0.5)}, \quad (13)$$

where TF is the term frequency, k_1 and b are the parameters, DL and AV DL 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 and. The SVM model is also a well-known learning method introduced by Cortes and Vapnik. Since the works of Joachims , researchers have successfully applied SVM to Various related tasks and presented some convincing results. The decision function in SVM is defined

Where x is the input space; b ϵ R is a threshold and

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

For the given training data

$$(x_i, y_i), \ldots, (x_l, y_l), \quad (15)$$

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

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

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 usingIn 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 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

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

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 method 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 Based Model with Deploying	Equation (9)
CBM Pattern Matching	Concept Based Model with Pattern matching	Equation (10)
nGram	nGram method with $n = 3$	3Gram
Rocchio	Rocchio method	Equation (11) $\alpha = 1, \beta = 0$
Prob	Probabilistic method	Equation (12)
TFIDF	TFIDF method	TFIDF Section VI-C.2
BM25	Probabilistic method	Equation (13) $k_1 = 1.2, b = 0.75$
SVM	Support Vector Machines method	Equation (14) $b = 0$

D.Hypotheses

The major objective of the experiments is to show how the proposed approach can help improving the effectiveness of Prototype-based approaches. Hence, to give a comprehen-sive investigation for the proposed model, our experiments involve comparing the performance of different Prototype-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 Prototype-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 discov-ered Models in text documents. This deploying approach is not only promising for Prototype-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 Prototype mining, sequential closed Prototype mining, frequent itemset mining, frequent closed itemset mining, where min sup = 0.2. The second category includes the concept-based model that uses the deploying method and the CBM Prototype

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

E. Experimental Results

This section presents the results for the evaluation of the proposed approach PTM (IPE), inner Prototype evolving in the Prototype taxonomy model. The results of overall comparisons are presented in Table 5, and the summarized results are described in Fig. 5. 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, item set-based data mining methods struggle in some topics as too Various 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.

The most important information revealed in this table is that our proposed PTM (IPE) outperforms not only the Prototype mining-based methods, but also the term-based methods including the state-of-the-art methods BM25 and SVM. PTM (IPE) also outperforms CBM Prototype 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(jT j jdj) 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.

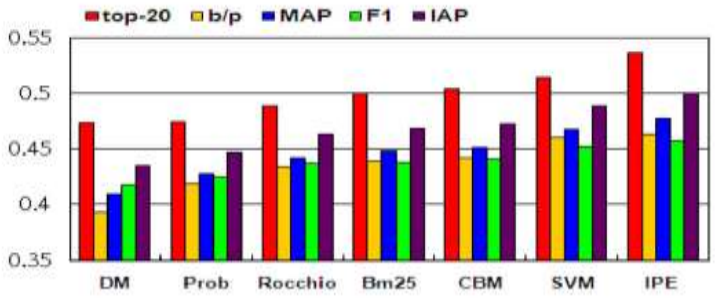

Fig. 5. Comparison of PTM (IPE) and other major models in five measures for the 100 topics.

TABLE 6
Performance of Inner Prototype Evolving in PTM on All Topics

	PDM	$IPE_{\mu=5}$
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**

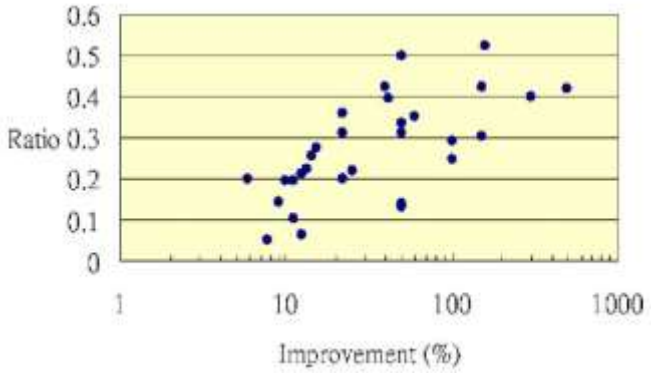

Fig. 6. The relationship between the proportion in number of negative documents greater than threshold to all documents and corresponding improvement on IPE with _ = 5 on improved topics.

F. Discussion

PDM to IPE: Table 6 depicts the figures of evaluating measures achieved by inner Prototype evolving methods (IPE) and pure Prototype 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:

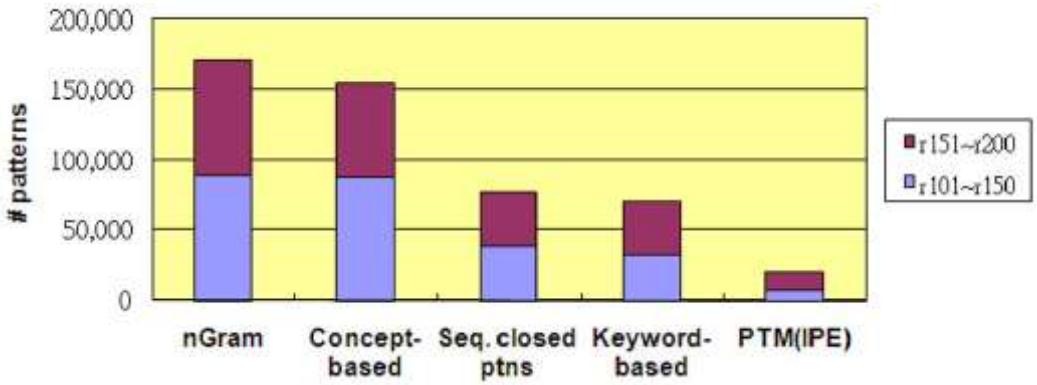

Fig. 7. Comparison in the number of Models used for training by each method on the first 50 topics (r101 r150) and the rest of the topics (r151 r200).

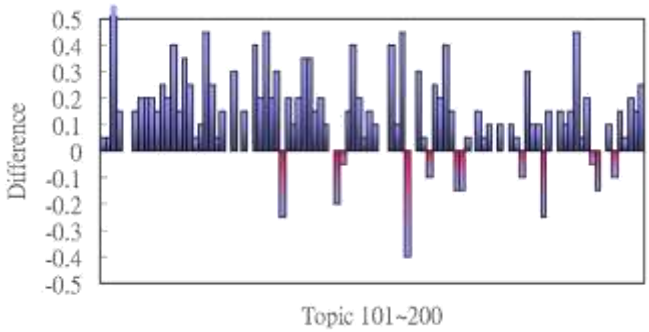

Fig. 8. Comparison of PTM (IPE) and TFIDF in top-20 precision.

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

The relationship of the improvement as inner evolving is applied and the abovementioned 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.

6.6.2 PTM (IPE) versus Other Models

The number of Models used for training by each method is shown in Fig. 7. The total number of Models 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 Models for concept learning compared to others. This is because the efficient scheme of Prototype pruning is applied to the PTM (IPE) method. Nevertheless, the classic methods such as Rocchio, Prob, and TFIDF adopt terms as Models in the feature space, they use much more Models than the proposed PTM (IPE) method and slightly less than the sequential closed Prototype mining method. Particularly, gram and the concept-based models are the methods with the lowest performance which requires more than 15,000 pat-terns for concept learning. In addition, the total number of Models 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 Models is obtained by using PTM (IPE) based on the first topics group

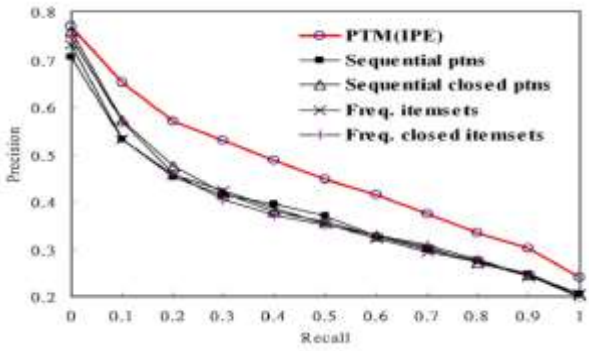

Fig. 9. Comparing PTM (IPE) with Data Mining methods on the first 50 TREC topics

A further investigation in the comparison of PTM (IPE) and TFIDF in top-20 precision on all RCV1 topics is depicted in Fig. 8. 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 behaviour 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 behaviours are found in topics r134 and r144.

The plotting of precisions on 11 standard points for PTM (IPE) and Prototype mining based methods on the first 50 topics is . The result supports the superiority of the PTM (IPE) method and highlights the importance of the adoption of proper Prototype deploying and Prototype evolving methods to a Prototype-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 Prototype 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 Prototype mining methods achieve similar results. However, the plotting curve for PTM (IPE) is much smoother than those for Prototype mining methods as there is no severe fluctuation on it. Another observation on this figure is that the Prototype 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 Prototype mining-based methods can improve the performance in the low-recall situation.

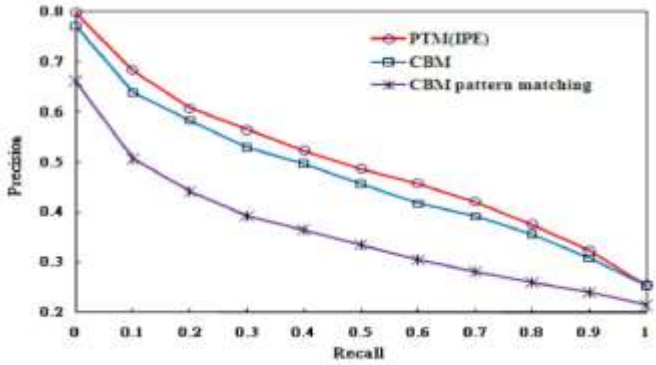

Fig. 10. Comparing PTM (IPE) with concept-based models on the all 100 TREC topics.

From the successful application of the proposed d-Models and inner Prototype evolving. The proper usage of d-Models, which has been proven previously, can overcome the misinterpretation problem and provide a feasible solution to effectively exploit the vast amount of Models 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 show in that the PTM (IPE) method is an ideal model for further developing Prototype mining based approaches.

As mentioned in the last section, PTM (IPE) is outperforms CBM Prototype Matching and CBM in all five 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. Fig. 10 shows the plotting of precision on 11 standard points for PTM (IPE), CBM, and CBM Prototype 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 Prototype Matching model. In general, the PTM (IPE) method outper-forms CBM in these experiments.

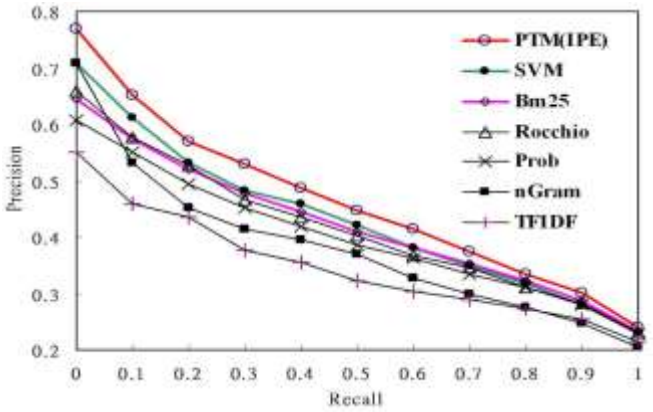

Fig. 11. Comparing PTM (IPE) with term-based methods on the first 50 TREC topics.

The 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 Fig. 9, 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 gram method achieves a noticeable score of precision at the first point where recall equals to zero, meaning that the n Gram method is able to promote top relevant documents toward the front of the ranking list. As mentioned before, data mining-based methods can perform well at the low-recall area, which can explain why n Gram has better results at this point. However, the scores for the n Gram method drop rapidly at the following couple of points. During that period, SVM, BM25, Rocchio, and Prob methods transcend the n Gram 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 n Gram method only at the last few recall points. In addition, the Prob method is superior to the n Gram method, but inferior to the Rocchio method.

VII. CONCLUSION

Various data mining techniques have been proposed in the last decade. These techniques include association rule mining, frequent item set mining, sequential Prototype mining, maximum Prototype mining, and closed Prototype mining. However, using these discovered knowledge in the field of text mining is difficult and ineffective. The reason is that some useful long Models with high specificity lack in support. That not all frequent short Models are useful. Hence, misinterpretations of Models derived from data mining techniques lead to the ineffective performance. In this research work, an effective Prototype discovery technique has been proposed to overcome the low-frequency and misinterpretation problems for text mining. The proposed technique uses two processes, Prototype deploying and Prototype evolving, to refine the discovered Models 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.

REFERENCES

[1] K. Aas and L. Eikvil, "*Text Categorisation: A Survey*," Technical Report Raport NR 941, Norwegian Computing Center, 1999.
[2] R. Agrawal and R. Srikant, "*Fast Algorithms for Mining Association Rules in Large Databases*," Proc. 20th Int'l Conf. Very Large Data Bases (VLDB '94), pp. 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*," Proc. IEEE Int'l Forum on Research and Technology Advances in Digital Libraries (ADL '98), pp. 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*," TREC, trec.nist.gov/ pubs/trec11/papers/kermit.ps.gz, 2002.
[6] N. Cancedda, E. Gaussier, C. Goutte, and J.-M. Renders, "*Word-Sequence Kernels*," J. Machine Learning Research, vol. 3, pp. 1059-1082, 2003.
[7] M.F. Caropreso, S. Matwin, and F. Sebastiani, "*Statistical Phrases in Automated Text Categorization*," Technical Report IEI-B4-07-2000, Instituto di Elaborazione dell'Informazione, 2000.
[8] C. Cortes and V. Vapnik, "*Support-Vector Networks*," Machine Learning, vol. 20, no. 3, pp. 273-297, 1995.
[9] S.T. Dumais, "*Improving the Retrieval of Information from External Sources," Behavior Research Methods*, Instruments, and Computers, vol. 23, no. 2, pp. 229-236, 1991.
[10] J. Han and K.C.-C. Chang, "*Data Mining for Web Intelligence*," Computer, vol. 35, no. 11, pp. 64-70, Nov. 2002.