Term Filtering with Bounded Error

Zi Yang, Wei Li, Jie Tang, and Juanzi Li
Tsinghua National Laboratory for Information Science and Technology
Department of Computer Science and Technology, Tsinghua University, China
$\{$*yangzi, tangjie, ljz*$\}$*@keg.cs.tsinghua.edu.cn, lwthucs@gmail.com*

***Abstract*—In this paper, we consider a novel problem referred to as term filtering with bounded error to reduce the term (feature) space by eliminating terms without (or with bounded) information loss. Different from existing works, the obtained term space provides a complete view of the original term space. More interestingly, several important questions can be answered such as: 1) how different terms interact with each other and 2) how the filtered terms can be represented by the other terms. We perform a theoretical investigation of the term filtering problem and link it to the Geometric Covering By Discs problem, and prove its NP-hardness. We present two novel approaches for both lossless and lossy term filtering with bounds on the introduced error. Experimental results on multiple text mining tasks validate the effectiveness of the proposed approaches.**

I. Introduction

In many text mining tasks such as text classification, text clustering, and information retrieval [1], [2], [3], each document is usually represented as a vector of terms. Such a presentation is effective in many applications. However, it also suffers from the possibly high computation-cost and intensive memory access, due to the large number of terms (features). For an example, Latent Dirichlet Allocation (LDA) [2] is a statistical topic modeling and has been successfully applied to various text mining applications. However, training an LDA model on a relative large data set takes days, even months [4]. Such a long running time significantly limits the use of LDA in real-world applications, in particular on the rapidly evolving social Web: the data set may be updated a few hours after an event bursts.

Different methods have been proposed to address the problem. Generally speaking, there are three trends for this purpose. The first one is to improve the algorithm itself, e.g., to reduce the computational complexity through special data structures or mathematic reduction. Another trend is to utilize modern high performance platforms, e.g., multi-core processors and distributed machines, to develop high performance implementations. Research efforts in this trend include parallel/distributed machine learning algorithms [5] using MPI and MapReduce [6]. The third one is to reduce the document collections using feature selection approaches, which tries to choose a subset of features and eliminate features of less importance [7].

In this paper, we propose the problem of *term filtering with bounded error*. The basic assumption is that each term captures more or less information and the goal is to obtain a subspace of features (terms) with a complete view of the original information space or with minimal information loss. The problem setting is similar, but different from, feature selection [7]. In both problems, metrics such as information gain or mutual informativeness are defined to measure each individual feature or the dependency to a group of features, and features with low scores in importance or high scores in redundancy will be removed. Since terms are usually correlated with each other, and a set of terms with low scores (e.g., information gain) do not necessarily mean they have a low informative degree when combining them together, in term filtering problem, we measure not only the informativeness of terms by considering the correlation between them, but also the information loss of individual document and the correlation degrees of documents. Term filtering functions as a preprocess, and we do not require any predefined tasks (e.g., document clustering, classification, retrieval, etc.) or labels for classification.

Moreover, we borrow the idea from data compression, especially, graph compression [8] if term document relation is represented equivalently as a bipartite graph. As an approach for preprocessing, the term filtering problem values the interpretability of filtered features (terms), completeness of the transformation from the originally features to the generated features (terms), and the recoverability, which are essential for various data mining tasks, e.g., document retrieval, and are easily ignored by traditional feature selection methods. Therefore, we quantify what the information loss is when filtering a set of terms to ensure the completeness of the collections. The importance of those characteristics in term filtering is demonstrated in Section V.

Equivalently, we have the following question: with a given *bounded error*, how many and which terms should be filtered? In this problem, we will quantitatively measure individual term/document and individual group of terms/documents by two user specified bounds.

The problem of *term filtering with bounded error* presents a set of unique challenges:

- *Information loss*. There might be (strong) correlations between terms. It is important to be able to quantify the information loss of filtering a single term/document and that of filtering a set of terms/documents.
- *Theoretical analysis*. Given a bounded error (a tolera-

tive information loss), which terms can be filtered? Is there always an exact solution? A thorough theoretical analysis to this problem is needed.

- *Efficient algorithm.* Real data sets are getting bigger with millions of documents. It is important to develop methods that can scale well to large data sets.

We aim to conduct a systematic study of the problem of term filtering with bounded error. Our work makes the following contributions. First, we formally define the problem and perform a theoretical investigation of this problem by linking it to the GEOMETRIC COVERING BY DISCS problem, and prove its NP-hardness. Second, we develop efficient algorithms for both lossless and lossy term filtering with bounds on the introduced error using locality-sensitive hashing (LSH). Third, the proposed approach is validated through an extensive set of experiments with multiple real-world text mining applications, which demonstrate the essential of term filtering problem.

The rest of the paper is organized as follows. In Section 2, we formally formulate the problem. In Section 3, we study a special case of the problem where the information loss is not allowed. In Section 4, we generalize the problem with positive errors and prove its NP-hardness. An approximate algorithm is proposed with a discussion on the efficiency and improvement of the algorithm. In Section 5, we compare the efficiency of the proposed parallel method and a randomized algorithm based on LSH, moreover we evaluate effectiveness of our proposed method.

II. PROBLEM DEFINITION

To quantify the loss when filtering a set of terms, we should not only consider the terms, but also consider the documents. This is because, filtering out a term may result in some information loss for the terms, but may have little impact on the document collection, i.e., introduce little change on the semantics of the collection. On the contrary, filtering out a set of terms in the same document, the meaning of the corresponding document may alter greatly. For example, if a set of typos are distributed in different locations in a document (*less correlated*), then we may be able to rectify each typo and understand the meaning of the document, but if multiple typos occur together in a key sentence (*more correlated*), then we are likely to misunderstand or even cannot understand its main idea. This inspires the definitions of the problem.

We denote the set of terms and documents by V and D, consider a collection of documents represented as a $|V| \times |D|$ co-occurrence matrix M, where each element $M_{ij} = n(w_i, d_j)$ denotes the number of times that term w_i occurs in document d_j; each row of matrix M corresponds to a vector presentation of word; and each column corresponds to a vector presentation of document. In term filtering, we usually need to group and evaluate the loss of a set of terms. For easy explanation, we introduce the notion of *superterm*.

Definition 2.1: (**Superterm**) A superterm s represents a new (virtual) term by grouping a set of terms together. Each superterm also has a vector representation, denoted as $\mathbf{s} = \{n(s, d_j)\}_j$, where $n(s, d_j)$ is the number of occurrences of superterm s in document d_j.

We define a mapping function $f(\cdot)$ to map terms to their superterms if they have been assigned to some superterms, or otherwise map to themselves. All the terms $\{w|f(w) = s\}$ take their common vector representation $\mathbf{s}$ instead of $\mathbf{w}$ after grouping. Moreover, we consider the problem in the context of text mining, thus all the terms and documents are represented in the non-negative vectors, i.e., it is defined in $\mathbb{R}^{|D|}$, where all the components are non-negative, i.e., $\mathbb{R}_+^{|V|} = \{\mathbf{x} \in \mathbb{R}^{|V|} | x_i \geq 0\}$ or $\mathbb{R}_+^{|D|} = \{\mathbf{x} \in \mathbb{R}^{|D|} | x_i \geq 0\}$.

Grouping and representing terms by superterms can reduce the term space. Our task then is to quantify the information loss in this process. For this purpose, some distances can be used to calculate the loss (error). Here we use a metric *Euclidean distance* and a pseudometric *cosine similarity* as examples:

$$d_E(\mathbf{x}, \mathbf{x}') = ||\mathbf{x} - \mathbf{x}'||_2 = \sqrt{\sum_{i=1}^{n} (\mathbf{x}_i - \mathbf{x}'_i)^2} \tag{1}$$

$$d_C(\mathbf{x}, \mathbf{x}') = \arccos \frac{\langle \mathbf{x}, \mathbf{x}' \rangle}{||\mathbf{x}|| \cdot ||\mathbf{x}'||} \tag{2}$$

where the inner product $\langle \cdot \rangle$ is defined as the dot product, and the norm $\|\cdot\|$ is the induced Euclidean norm. Strictly, d_C is not a metric, but it can be decomposed into two functions: $d_C(\mathbf{x}, \mathbf{x}') = d_C|_{S^{n-1}}(u(\mathbf{x}), u(\mathbf{x}'))$, where $u(\mathbf{x}) = \frac{\mathbf{x}}{||\mathbf{x}||}$ is a normalization function, and thus $d_C|_{S^{n-1}}$ is a metric restricted on $(n-1)$-dimensional sphere S^{n-1}.

Based on the distance metrics, we can define the error (information loss) in the term filtering process.

Definition 2.2: (**Information Loss**) If a superterm s is introduced to substitute a set of terms $\{w_i\}$, (or $\{w_i\}$ are grouped into s), then the error of the term w is defined as:

$$\texttt{error}_V(w) = d(\mathbf{w}, \mathbf{s}) \tag{3}$$

where $d(\cdot, \cdot)$ is Euclidean distance or cosine similarity defined on $\mathbb{R}_+^{|V|}$.

Analogously, document d, if satisfying $n(s, d) > 0$, will have a different representation $\mathbf{d}'$, thus we define the error of a document as:

$$\texttt{error}_D(d) = d(\mathbf{d}, \mathbf{d}') \tag{4}$$

where $d(\cdot, \cdot)$ is Euclidean distance or cosine similarity defined on $\mathbb{R}_+^{|D|}$.

Now, we need to specify the vector representation $\mathbf{d}'$ for documents after grouping. The vector representation for document d_j is in turn represented by superterms $s = f(w_i)$

instead of original terms w_i. More specifically, the i-th component in vector representation for document d_j is given by: $n(w_i, d_j)' = n(s, d_j)$. In the next section, we will discuss how to obtain the vector representation $\mathbf{s}$.

Based on the above concepts, we can formally define the task of term filtering with bounded error.

Definition 2.3: (**Term Filtering with Bounded Error**) Given a vocabulary V, a document collection D defined on it, an association function $n : V \times D \to \mathbb{N}$, and non-negative numbers ϵ_V and ϵ_D. The problem is is to minimize the size of superterms $|S|$ subject to the following constraints:

(1) $f(V) = \amalg_{s \in S} s$,

(2) for all $w \in V$, $\texttt{error}_V(w) \leq \epsilon_V$,

(3) for all $d \in D$, $\texttt{error}_D(d) \leq \epsilon_D$.

The first constraint describes that each term should be grouped into exactly one superterm, and second and third constraints restrict the errors for all the terms and documents are bounded by the specified errors. Note that this is still a general formulation. In the following sections, we will further specialize the problem with different configurations and propose efficient algorithms to solve it.

Our formulation of term filtering with bounded error is quite different from existing works on feature (term) selection. Yang and Wilbur [9] present a stop word identification method to automatically generate domain specific stoplists. Li and Sun [10] propose a method for scalable term selection by separately measuring the discriminability and coverage of the selected term set. Yang and Pedersen [11] have a comparative study on existing methods for term selection. Most existing methods for feature selection separately consider the informativeness of each term and ignore correlation between terms, and no error bounds on the loss of information could be specified and guaranteed.

III. Lossless Term Filtering

Theoretical Basics In the Term Filtering with Bounded Error problem, if we set $\epsilon_V = 0$ and $\epsilon_D = 0$, in the purpose of filtering terms without information loss of any term and document, then this sub-problem is called the Lossless Term Filtering problem. Intuitively, this problem could be solved by simply grouping the terms with the same vector representation. Formally, we have the following theorem.

Theorem 3.1: If the parameters ϵ_V and ϵ_D in the Term Filtering with Bounded Error problem are set 0, then the exact optimal solution of the problem is yielded by grouping terms of the same vector representation.

Before we give the proof to the theorem, we consider the information loss of terms and documents at the beginning, and give a necessary and sufficient condition for that the variation of errors of all the terms and documents remains 0. Note that if all $\{w\}$ have the same representation, the superterm s is intuitively chosen as any term w that is grouped into s, i.e., $\mathbf{s} = \mathbf{w}$.

Algorithm 1: Lossless Term Filter: LTF

```
input  : A $|V| \times |D|$ co-occurrence matrix $M$
output : A set $S$ of superterms $\{s_i\}_i$, such that $\amalg_i S_i = V$
1  initialize $S = \phi$, $U = \phi$, and mapping $f(V) = \phi$;
2  foreach document $d \in D$ do
3      $\hat{S} = \{\hat{s_i}\}_i$, each contains terms of the same occurrence in $d$;
4      foreach local superterm $\hat{s_i} = \{(w_{ij}, v_{ij})\}_j \in \hat{S}$ do
5          foreach superterm $s_k$ in $f(\{w_{ij}\}_j) \setminus U$ do
6              $s^* = \{w \in \{w_{ij}\}_j | f(w) = s_k\}$;
7              if $s_k = \phi$ then ADD ($s^*$);
8              else
9                  if $|s^*| > 1$ then $S \leftarrow S \cup s^* \setminus s_k$, $f(s^*) \leftarrow s^*$;
10                 else $S \leftarrow S \setminus s_k$;
11                 ADD ($s_k \setminus s^*$);
12 ADD ($s$): if $|s| > 1$ then $S \leftarrow S \cup s$, $f(s) \leftarrow s$ else $U \leftarrow U \cup s$;
```

Lemma 3.1: When a new superterm s is introduced to replace a set of superterms $\{w_i\}_i$, the errors of all the terms and documents equal to 0, if and only if every term in s has the same vector representation.

The proof of the lemma is given in the Appendix. Note that if the error bound ϵ_V is defined by cosine similarity, then two proportional terms are certainly grouped without any loss on the terms. If the information loss on the set of terms is considered, documents are indeed affected by grouping such terms. It manifests the motivation of the problem, and two different types of vector representations with two error bounds are introduced. Therefore the theorem is proved based on the lemma.

Lossless Term Filter Algorithm Due to the large memory requirement for storing the vector representations, we do not directly compare each pair of terms, in turn we dynamically maintain a superterm set S during processing the document collections, which is depicted in Algorithm 1. Specifically, when inputting document d, we first locally find a set of "local" superterms, denoted by $\hat{s}$, by grouping the terms with the same number of occurrences in this document (line 3). For the terms in each "local" superterm, we find their superterms by invoking f (line 5). Let s^* denote newly generated superterm. If the terms are contained in neither S nor U, they are grouped into s^* and added to S (line 7), otherwise s_k will be split into s^* and $s_k \setminus s^*$ since they exhibit differently on document d, and both subsets are then checked and added to S (line 9 - 11).

At the time that k documents ($k < |D|$) have been processed, the superterm set S is consistent with the first k documents. Hence, as some new superterms are continually added to S, some others will be removed from S and put into U, which contains terms that are not grouped in any superterm since they are inconsistent with the first k documents.

Though Lossless Filtering is a special (and simple) case of Term Filtering with Bounded Error problem, it plays an important role to reduce the size of the vocabulary, in particular on real large-scale document collections. Our preliminary analysis on a large online encyclopedia data set Baidu Baike[1] (containing 1,531,215 documents) shows that the vocabulary size could be reduced from 1,522,576 to 714,392 by only applying the lossless filter, i.e., we can obtain a $> 50\%$ reduction of the term vocabulary.

IV. Lossy Term Filtering

NP-hardness of the Problem If $\epsilon_V > 0$, $\epsilon_D > 0$, and the numbers of documents and terms are sufficiently large, the sub-problem is called as Term Filtering with Bounded Error problem. For this problem, we have the following NP-hardness results, and the proof of the theorem is given in the Appendix.

Theorem 4.1: (**NP-hardness**) The Term Filtering with Bounded Error problem with $\epsilon_V > 0$ and $\epsilon_D > 0$ is NP-hard if the metric for calculating the errors is defined as Euclidean distance and the number of documents $|D| \geq 2$.

Greedy Algorithm for Lossy Term Filter We develop a greedy algorithm to solve this problem. The basic idea is to replace the original space of terms with a reduced space of superterms. More specifically, we assume that each term can be represented by a superterm (or itself) with a certain loss. The superterm is formed by grouping a set of terms. We employ two practical methods to define the vector representation of the superterm.

Winner-take-all: Select the most representative term contained in the grouped terms to represent the superterm s. For simplicity, we define

$$i^* = \underset{i}{\operatorname{argmin}} \sum_j ||\mathbf{w_i} - \mathbf{w_j}||_2^2 \tag{5}$$

where i and j run over all the indexes of terms grouped into s, and $\mathbf{s} = \mathbf{w_{i^*}}$.

Average-occurrence: Use the average number of occurrence of all the grouped terms to represent the superterm, i.e., $\mathbf{s} = \frac{1}{n}\sum_i \mathbf{w_i}$, n is the number of terms grouped into s, and i runs over all the indexes of such terms.

We can take a similar method to further group superterms to obtain the superterms' superterms. In essence, the Term Filtering with Bounded Error problem can be considered as a partition problem, in which each object is restricted to appear in one of the clusters. Traditional greedy algorithms transform partition problems into set cover or dominating set problems, and achieve $O(\log(n))$-approximation in a Greedy Cover-style, e.g., [12]. However, in our problem, the choice of a superterm may change the validity of other candidate superterms due to the additional constraint ϵ_D for documents.

Algorithm 2: Greedy Lossy Term Filter: `Greedy LF`

```
input  : A $|V| \times |D|$ co-occurrence matrix $M$, error bounds $\epsilon_V > 0$ and
         $\epsilon_D > 0$ for terms and documents, which is processed by LTF
output : A set $S$ of superterms $\{s_i\}_i$, such that $f(V) = \amalg_{s \in S} s$
   // initialize the candidate set
1  $S \leftarrow \phi$, $T \leftarrow \phi$, $G_i \leftarrow \phi$, $H_i \leftarrow \phi$;
2  foreach term pair $w_i, w_j \in V$ and $d(\mathbf{w_i}, \mathbf{w_j}) \leq \epsilon_V$ do
3      $G_i \leftarrow G_i \cup \{w_j\}$, $G_j \leftarrow G_j \cup \{w_i\}$;
4  UPDATE($\hat{S}$);
   // group vectors
5  while $\exists i$ s.t. $H_i \neq \phi$ do
6      $i^* \leftarrow \arg\max_i |H_i|, S \leftarrow S \cup \{H_{i^*}\}, T \leftarrow T \cup H_{i^*}$;
7      foreach term $w_j \in H_{i^*}$ do
8          $G_j \leftarrow \phi$, $H_j \leftarrow \phi$;
9          foreach term $w_i \in V \setminus T$ do $G_i \leftarrow G_i \setminus \{w_j\}$;
10     UPDATE($As(As(H_{i^*}))$);
11 UPDATE ($S$): foreach term $w_i \in S$ do
12     $s \leftarrow \{w_i\}$;
13     while $G_i \neq \phi$ do
14         $w_{j^*} \leftarrow \arg\max_{w_j \in G_i} d(w_i, w_j)$, $G[w_i] \leftarrow G[w_i] \setminus \{w_j^*\}$;
15         foreach document $d \in As(s \cup \{w_{j^*}\})$ and $\mathrm{error}_D(d) \leq \epsilon_D$ do
16             $s \leftarrow s \cup \{w_{j^*}\}$, $H_i \leftarrow H_i \cup \{w_{j^*}\}$;
```

Based on the above consideration, we develop a greedy algorithm (as summarized in Algorithm 2) for the Term Filtering with Bounded Error problem in a Winner-take-all fashion. Firstly, we only consider the constraint given by ϵ_V. The similarities between terms are calculated beforehand and remain the same during clustering, and then a set of candidate "superterms", denoted by G, are generated. Secondly, the validity of the candidate superterms, denoted by H in the Algorithm 2, should be confirmed repeatedly if ϵ_D is also considered. Specifically, we first invoke the lossless term filter algorithm (line 1) to preprocess the input matrix M. Then we calculate the pair-wise similarities, and those less than ϵ_V are added into G (line 3). For each term w_i, we successively put terms into the group H_i of w_i, starting from the most "similar" term w_{j^*} (line 14), and try to find such terms as many as possible under the condition that ϵ_D is satisfied for all related documents (line 15 - 16). Finally the first "winner" w_{i^*} is the term whose group contains the most terms (line 6), and the first superterm is generated subsequently. As described above, related similarity scores and the candidate sets should be updated (line 7 - 10) and more superterms are then chosen according to the similar scheme. In Algorithm 2, $As(w) = \{d | n(w, d) > 0\}$ and $As(d) = \{w | n(w, d) > 0\}$.

The complexity of lossy filter is $O(|V|^2 + |V| \cdot T \cdot |D| + T'(|V|^2 + T \cdot |D|))$, where T, T' are the number of iterations in Algorithm 2 and $T, T' \ll |V|$. Additionally, the algorithm is easy to parallelize by executing line 12 - 16 of Algorithm 2 with multiple threads (or on multi-cores), which reduces the complexity to $O(|V|^2 + (|V| \cdot T \cdot |D| + T'(|V|^2 + T \cdot |D|))/P)$, where P is the number of threads. We implement the parallel algorithm and evaluate the speed-up in the experiment.

Efficient Candidate Superterm Generation From the above discussion on complexity, we see that Algorithm 2 is still computationally intensive and the major computational cost lies in the calculation of the pair-wise distance between terms. To further improve the efficiency, we leverage the

[1]http://baike.baidu.com

technique of locality-sensitive hashing (LSH) to reduce the candidate superterm space.

LSH [13] is a randomized approach to quickly find items from a large set closest to a queried item. The basic idea behind LSH is that if two data points have a same hash value with several randomized hash projections, then the two points are close to each in the original space. LSH does not guarantee an exact answer but instead provides a high probability guarantee that it will return the correct answer or one close to it, which is known as the locality-sensitive properties.

To apply LSH in our problem, we need first define a family of hash functions mapping the term vector space $\mathbb{R}^{|D|}$ to a discrete vector space embedded in $\mathbb{R}^h$ and $h \ll |D|$. Studies have shown that there are several LSH families preserving the locality-sensitive properties. For the Euclidean distance, the family of hash mappings is chosen as the set of independently random projections onto a 1-dimensional line, and then the line is chopped into segments plus a shift value [14]. Formally,

$$h_{r,b}(\mathbf{w}) = \left\lfloor \frac{\mathbf{r} \cdot \mathbf{w} + b}{w(\epsilon_V)} \right\rfloor \tag{6}$$

where $\mathbf{r}$ is the projection vector, whose components are independently drawn from the Gaussian distribution. Parameter $w(\epsilon_V)$ is the width of the buckets. For cosine distance, a random unit-length vector $\mathbf{r} \in \mathbb{R}^{|D|}$ is picked and the mapping is defined as

$$h_r(\mathbf{w}) = \operatorname{sign}(\mathbf{r} \cdot \mathbf{w}) \tag{7}$$

Based on the mapping function, we can replace the original candidate superterm generation (lines 2 in Algorithm 2) with a new procedure, as shown in Algorithm 3. The key set of G contains the term w_{i^*} that co-occurrences in hash buckets with the most other terms, then G_{i^*} could be directly determined by search ϵ_V-near neighbors in all the buckets. We note that since G does not need maintaining, and is re-generated at each iteration when a new superterm is formed, G contains only one element. However, our preliminary experiments show that in the subsequent iterations, the superterms are combinations of a very few local terms, and thus the generated mappings could be reused in several following iterations, in other words, G contains more than one superterms, which could improve the performance.

Time complexity The time complexity can be easily calculated as follows:

$$|V| \cdot |L| + |V| \cdot \max_{v \in V} |\{w \in V | d(v,w) < \epsilon_V)\}| \tag{8}$$

where $|L|$ is the number of generated lines. As we know that $\max_{v \in V} |\{w \in V | d(v,w) < \epsilon_V)\}|$ is less than n, we can expect a faster running time, i.e., $\ll |V|^2$.

Algorithm 3: LSH-based Candidate Superterm Generation

```
1 Generate set of lines $L = \{(\mathbf{r_i}, b_i\}_i$, and initialize buckets $B_{ik} \leftarrow \phi$;
2 foreach $w_j \in V$ and $(\mathbf{r_i}, b_i) \in L$ do
3     Calculate bucket index $k_{ij} = h_{r_i,b_i}(\mathbf{w_j})$ according to Eq. 6;
      $B_{ik_{ij}} \leftarrow B_{ik_{ij}} \cup \{w_j\}$;
4 foreach $(\mathbf{r_i}, b_i) \in L$ and $w_j \in V$ do
5     $N_j \leftarrow \sum_i |B_{ik_{ij}}|$; $G_j \leftarrow \cup_i B_{ik_{ij}}$;
6 $V' \leftarrow$ Sort $V$ in the ascending order of $\{N_j\}_j$;
7 foreach $w_i \in V$ and $w_j \in G_i$ do
8     if $d(w_i, w_j) \geq \epsilon_V$ then $G_i \leftarrow G_i \setminus \{w_j\}$;
9     $H_i \leftarrow G_i$;
```

V. Experiments

A. Settings

1) Data Set: We conduct the experiment on two real-world data sets: Academic and 20-Newsgroups. ArnetMiner[2] [15] is an academic publication collection, which contains 10,768 papers and has 8,212 terms. 20-Newsgroups[3], is a widely used data collection, which consists of 18,774 postings form 20 Usenet newsgroups, and has 61,188 terms. Preprocessing includes stopword filtering and word stemming. All data sets, codes, and detailed results are publicly available[4].

2) Baseline Method: We compare the proposed method with four pervasively applied methods. Document frequency criterion, term strength, sparse principal component analysis, are state-of-art task-irrelavent methods for feature selections in text mining, and Chi-statistic is considered effective in feature selection for supervised classification tasks [11].

DF Document frequency criterion has been shown as one of the most effective feature selection methods for text. We define a frequency-based baseline method based on the approach from [9], which filters words whose document frequencies are less or more than a threshold. Here the threshold is specified such that the same ratios of terms are filtered as in our method.

TS Term strength [16] evaluates term importance based on how commonly a term is likely to appear in "closely-related" documents. The document similarity is measured by the cosine similarity, and two documents are closely-related if the cosine value of their vector representations is greater than 0.01. The importances of terms are estimated by the following equation.

$$\text{TS}(w) = \frac{\text{\# of pairs where } w \text{ co-occurs in both documents}}{\text{\# of pairs where } w \text{ co-occurs in either document}} \tag{9}$$

The threshold for filtering terms is also determined such that the same ratios of terms are filtered as in our method.

SPCA Sparse principal component analysis [17] extends principal component analysis, a commonly used approach

[2]http://www.arnetminer.org/download/

[3]http://people.csail.mit.edu/˜jrennie/20Newsgroups/

[4]http://www.arnetminer.org/termfilt/

for dimension reduction, by constraining that there are only a few nonzero coefficients in the principal components for easy interpreting. We apply the greedy method for an approximate solution of SPCA with a cardinal penalty [18]. Specifically, SPCA finds sparse factor vector $\mathbf{z}$ which explains a maximum amount of variance, and it can be written as:

$$\max_{||\mathbf{z}||\leq 1} \mathbf{z}^\top \Sigma \mathbf{z} - \rho \text{Card}(\mathbf{z}) \tag{10}$$

where $\rho > 0$ is a parameter that controls sparsity, and $\text{Card}(\mathbf{z})$ is the number of nonzero coefficients of $\mathbf{z}$.

To compare with the filtered document collection with k unique terms, we look for a sparse coefficient vector $\mathbf{z}$ with k non-zero components, whose i-th component is an indicator of whether w_i is filtered or not. Details of the algorithm for solving SPCA can be referred to [18].

CHIMAX Chi-statistic is a state-of-the-art "supervised" approach for feature selection in various classification tasks. For evaluating the effectiveness of each individual feature, it measures the lack of independence between a term and the category. In CHIMAX criterion, terms that take the maximum on all categories are selected. Note that the problem of term filtering differentiates from feature selection in that term filtering aims to reduce the vocabulary without labeled information or any predefined task, and thus, we only perform CHIMAX on 20-Newsgroups data set.

B. Filtering Results and Evaluation of Information Loss

For our proposed algorithm, we use the Euclidean metric, and set ϵ_V from 1 to 2.5 on ArnetMiner data set and from 2 to 3.5 on 20-Newsgroups data set. Moreover, the lossless filter is also compared. To justify the motivation of our work, we gradually decrease the tolerable information loss of documents, i.e., ϵ_D is decreased from ϵ_V to 1 (for ArnetMiner) or to 2 (for 20-Newsgroups). The ArnetMiner data set is filtered using the parallel version of Greedy Lossy Filter, and 20-Newsgroups data set cannot be filtered efficiently even with the parallel filter. It is then filtered with the LSH version.

Filtering results The number of terms drops from the original 8212 down to 1694 on ArnetMiner data set if our method is applied with $\epsilon_V = \epsilon_D = 2.5$, which is shown in Table I. It means approximately 80% of the memory allocation is saved for the co-occurrence matrix M and the overall information loss is controlled by user specified errors. Since different data sets have different degree of information redundancy, when setting $\epsilon_V = 3.5$ on 20-Newsgroups data set, the size of vocabulary could drop to 59.6% of the original collection.

Evaluation of information loss We obtain five filtered document collections on each data set respectively by applying our proposed methods and the baseline methods DF, TS, SPCA on ArnetMiner data set. And further, we could evaluate the information loss in terms of error. Specifically, the error of terms and documents are defined as in Section II. The result is shown in Figure 1.

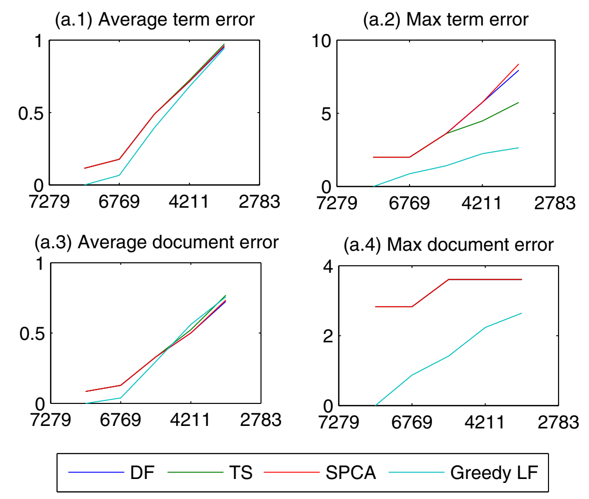
Figure 1(a) compares the errors of our proposed methods and the baseline methods when setting $\epsilon_D = \epsilon_V$ in our proposed model. We respectively compare the average error of all terms/documents and the maximum error of terms/documents, and we can see from Figure 1(a.1) and Figure 1(a.3) that the document collection filtered by our proposed method achieve less loss of information in term of term/document distance on average. More importantly, our proposed method could constrain the maximum error of terms and documents while filtering, which is clearly shown in Figure 1(a.2) and Figure 1(a.4). In Figure 1(b), we fix $\epsilon_V = 2.5$ and vary ϵ_D from 2.5 to 1.0. We could see that the maximum error of terms and documents are as well bounded by the fixed ϵ_V and tuned ϵ_D, and the average error of document synchronously decreases. Note that we do not explicitly constrain the error of terms ϵ_V, however, it drops from 0.757 to 0.502 to adapt a tighter bound of ϵ_D.

C. Efficiency Performance of Parallel Algorithm and LSH

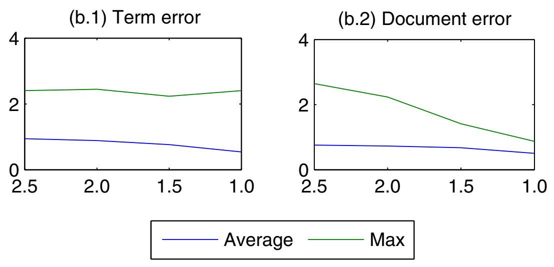
The algorithm for lossless term filtering is very efficient, while the algorithm for lossy term filtering is relatively computationally costly. To accelerate the algorithm, we develop a parallel implementation of the algorithm (Algorithm 2) and further improve the process of generating candidate superterms through a randomized algorithm LSH (Algorithm 3). To quantitatively evaluate our method, we present the

Table I
RESULT OF TERM FILTERING

	ϵ_V	ϵ_D	**# terms**	**Term ratio**
ArnetMiner	Original		8,212	100%
	Lossless filter		7,279	88.6%
	1	1	6,769	82.4%
	1.5	1.5	4,211	51.3%
		1	4,231	51.5%
	2	2	2,783	33.9%
		1.5	2,931	35.7%
		1	3,472	42.3%
	2.5	2.5	1,694	20.6%
		2	1,856	22.6%
		1.5	2,569	31.3%
		1	3,293	40.1%
20-Newsgroups	Original		61,188	100%
	Lossless filter		53,406	87.3%
	2	2	47,560	77.7%
	2.5	2.5	40,791	66.7%
		2	41,941	68.5%
	3	3	35,914	58.7%
		2.5	37,451	61.2%
		2	39,202	64.1%
	3.5	3.5	33,699	55.1%
		3	35,089	57.3%
		2.5	35,753	58.4%
		2	36,479	59.6%



(a) The errors of terms and documents when setting $\epsilon_D = \epsilon_V$



(b) The errors of terms and documents when setting $\epsilon_V = 2.5$ and varying ϵ_D from 1 to 2.5

Figure 1. Evaluation of information loss

efficiency performance of both enhanced algorithms.

We test the parallel algorithm on an 8-core computer by setting the number of threads as 4 and 8, and the absolute time varies from 40s to 2135s for 4 threads and varies from 20s to 1047s for 8 threads. For LSH, parameter w is set as $100 \times \epsilon_V^{-1}$, and the hash family contains 10 hashing mappings. In Figure 2, we list the absolute running time (in seconds) of the parallel algorithm and LSH algorithm on ArnetMiner data set. Note that with ϵ_V and ϵ_D changed from 1 to 2.5, the amount of candidate pairs increases and the calculations are more effectively distributed in parallel. The time cost of LSH varies from 65s to 3s on the same data set. In our experiments, we find that LSH favors moderate error bounds for most data collections (e.g., $1.5 < \epsilon_V, \epsilon_D < 3$ on ArnetMiner). A tight error bound may result in continuously generating small candidate superterms, and thus increase the iterative running time for generating superterms. It is the same with a tight bound configured after several iterations.

D. Applications

We present the results in three typical data mining applications: document clustering, classification, and information retrieval. In each application, we apply the proposed term filtering algorithm to filter terms within a bounded error, and then evaluate the accuracy loss and efficiency gain, by comparing with the baseline methods.

Document Clustering We use the ArnetMiner data set for this application. For the clustering algorithm, we use the probabilistic clustering method Latent Dirichlet Allocation (LDA) [2], and use perplexity as the evaluation measure [19], specifically,

$$P = \exp\left[\frac{\sum_{d=1}^{|D|}\sum_{w=1}^{|V|} n(w,z)\cdot\log\left(\sum_{z=1}^{|T|}\phi_{\mathbf{z}w}\theta_{\mathbf{d}z}\right)}{\sum_{d=1}^{|D|}\sum_{w=1}^{|V|} n(w,z)}\right] \quad (11)$$

where $n(w,z)$ is the number of times that term w is assigned topic z, and $\phi_{\mathbf{z}w}$ and $\theta_{\mathbf{d}z}$ are the w-th and z-th components in $\phi_{\mathbf{z}}$ and $\theta_{\mathbf{d}}$. Higher values of perplexity indicate a higher misrepresentation of the words of the test documents by the trained topics.

In the experiment, we set the number of topics $|T| = 80$, hyperparameters $\alpha = 0.65, \beta = 0.1$, and the number of sampling iterations as 1000. We also compare with the result on the original document collection. In all the six experiments, the document collections are split into training set (80%) and test set (20%).

Figure 3 shows the comparison of our method with the baseline methods in terms of perplexity. We can see that the perplexity of document collections derived by our method decreases from 844 to 147 as more terms are filtered out, which is much lower than those applying baseline methods in all cases. When ϵ_V is fixed, and ϵ_D is tuned from 1 to ϵ_V, we find that perplexity also increases. It is because that conventional feature selection methods are usually designed for achieving better classification accuracy, and it cannot properly handle clustering problems, esp., document clustering. Moreover, instead of focusing on selecting a set of features, and ignoring the rest features, our method can take the advantage of those less informative features in clustering.

Document Classification For classification, we choose the first 10 categories (18,774 postings) from the labeled 20-Newsgroups data set, which is then split into training set (11,269 documents) and test set (7,505 documents). We use LIBSVM with linear kernel as the classification method[5]. We use accuracy as the evaluation measure.

The accuracy of predictions on different filtered collections is shown in Figure 4. We can see that in this application, all the methods exhibit similarly with no more than 0.4% difference when only a few terms are filtered out. In comparison, our method, with DF and SPCA, nearly represents the classification ability of the original model when large amount of terms are filtered out, and the accuracy varies in a small interval between 77.10% and 77.70%. This indicates that our method perfectly retains information

[5]http://www.csie.ntu.edu.tw/~cjlin/libsvm

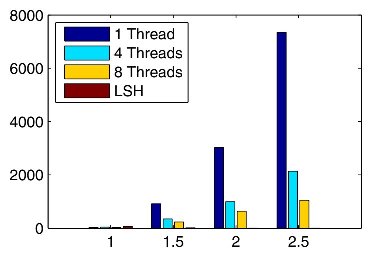


Figure 2. Performance of speed-up

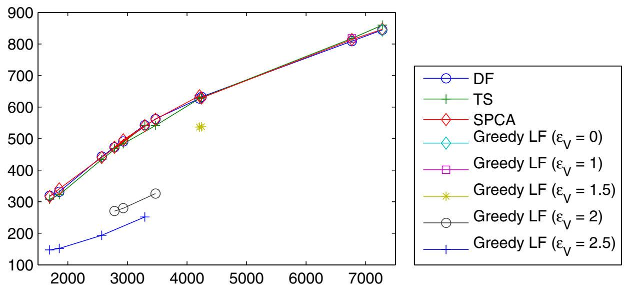


Figure 3. Result of clustering perplexity

within the given information loss. CHIMAX method could achieve better performance when fewer terms are filtered, and it drops by nearly 6% if more terms are filtered.

Document Retrieval Another practical application is document retrieval [20]. We use ArnetMiner data set for document retrieval. Documents and queries are represented in vectors, and ranking result is obtained by using language model for IR [21]. Details can be referred to [22].

We collect two types of queries (14 in total), 7 informational queries (e.g., "information extraction") and 7 navigational queries (e.g., "ObjectRank" and "SCRFs", which are names of algorithms). The ground truth data for informational queries are pooled from manual annotations, and the documents for retrieving navigational queries are just the few documents containing the query terms, followed by sorting the documents in ascending order of the dates of publishing.

The results are shown in Figure 5. We see that our method (Greedy LF) clearly outperforms the baseline methods. We have also found that for the navigational queries, most baselines fail to retrieve relevant documents. It is because that for the navigational queries, the baseline methods hurt the retrieval performance, by simply filtering less informative terms. In contrast, our method considers the error of each individual terms, and thus all the terms (e.g., "information extraction" or "ObjectRank") can be treated equally.

VI. Related Work

Feature Selection The most relevant work to ours is feature selection [23]. To apply feature selection methods on term filtering, they usually focus on removing non-informative terms according to corpus statistics [9].

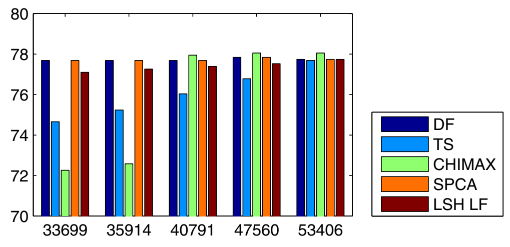


Figure 4. Result of classification accuracy

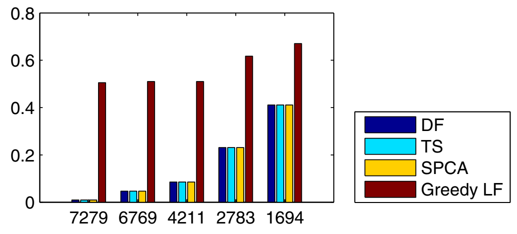


Figure 5. Result of document retrieval MAP

Some feature selection methods, e.g., information gain [24] or mutual information [25], are task-sensitive, i.e., category information of the terms should be provided in the training set, which is used as a clue to group the terms. Besides requiring the categorization information, some feature selection methods assess subsets of features according to their usefulness to a given follow-up learner. Document frequency and term strength [16] are the two widely used task-irrelevant criteria for feature selection. Document frequency criterion filters out the terms whose document frequency is less than some predetermined threshold. It can naturally scale to very large corpora. However, it is based on the assumption that low-DF terms are relatively informative [11]. Term strength [16] considers the correlations of documents, and it evaluates term importance based on how commonly a term is likely to appear in "closely-related" documents, where a threshold for measuring the document similarity, and another threshold for filtering terms should be predetermined. Normally, they are estimated experimentally, and they are not referential to different domains. Empirical comparisons of feature selection methods can be referred to [11] and [26]. Our work explicitly considers the information loss of a set of terms by inspecting the loss of documents, and thresholds can be specified by the required error bounds.

Some dimension reduction methods are proposed for reducing the size of input space, e.g., [27] also consider information loss, e.g., the change of mutual information between variables. The best (in mean-square sense) and most widely used way to do this is principal component analysis (PCA), however, it suffers from high time cost and memory requirement. Roweis [28] proposes for finding only a few eigenvectors and eigenvalues of a large matrix. Bingham and Mannila [29] apply PCA to dimension reduction in text corpora. Some classification methods, e.g., SVM, C4.5, etc. have also been employed for dealing with redundant features in text mining. e.g., [30] and [31]. Different from various term reduction approaches, the reduced low-dimensional feature space could hardly be explicated as a text collection any longer, and thus it restrains the range of application.

Locality-sensitive Hashing Locality-sensitive hashing [13] has been designed for fast R-near neighbor problem search. A number of works discuss the choice of hash functions for different distance metrics, or the LSH families to ensure the locality-sensitive properties are held, e.g., Hamming distance [13], l_s distance [14], cosine similarity [32], and a survey of those LSH families could be found in [33]. It has been applied to various problems, including document clustering and retrieval. Haveliwala et at. [34] use LSH for document clustering and fast retrieval on the Web. Ravichandran et al. [35] apply LSH for noun clustering by employing LSH families for cosine similarity.

VII. Conclusion

In this paper, we propose the problem of *Term Filtering with Bounded Error*. We formally define the problem and perform a theoretical investigation of this problem, and prove its NP-hardness. We develop two approaches for both lossless and lossy term filtering with bounds on the introduced error. Experimental results on several typical text mining tasks validate the effectiveness of the proposed approach. With a small introduce error (e.g., $\epsilon_V = 1$ and $\epsilon_D = 1$), our algorithm can significantly reduce the original term space (15% of the original space). We further validate the effectiveness of our approach on three real-world text mining applications.

In the future, we plan to put forward algorithms with heuristics for specific text mining tasks. Moreover, it is also interesting to implement the method on a distributed platform for dealing with real large data.

VIII. *ACKNOWLEDGMENTS

The work is supported by the NSFC (No. 61073073, No. 60703059), NSFC Key Fund (No. 60933013, No.61035004), National High-tech R&D Program (No. 2009AA01Z138).

References

[1] A. Dasgupta, P. Drineas, B. Harb, V. Josifovski, and M. W. Mahoney, "Feature selection methods for text classification," in *KDD'07*, 2007, pp. 230–239.

[2] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.

[3] X. Wei and W. B. Croft, "Lda-based document models for ad-hoc retrieval," in *SIGIR '06*, 2006, pp. 178–185.

[4] D. Newman, A. Asuncion, P. Smyth, and M. Welling, "Distributed inference for latent dirichlet allocation," in *Proceedings of NIPS'07*, 2007.

[5] C.-T. Chu, S. K. Kim, Y.-A. Lin, Y. Yu, G. Bradski, A. Y. Ng, and K. Olukotun, "Map-reduce for machine learning on multicore," in *NIPS '06*, 2006, pp. 281–288.

[6] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," in *OSDI '04*, 2004, pp. 10–10.

[7] L. Yi, "Web page cleaning for web mining through feature weighting," in *IJCAI '03*, 2003, pp. 43–50.

[8] S. Navlakha, R. Rastogi, and N. Shrivastava, "Graph summarization with bounded error," in *SIGMOD '08*, 2008, pp. 419–432.

[9] Y. Yang and J. Wilbur, "Using corpus statistics to remove redundant words in text categorization," *Journal of the American Society for Information Science*, vol. 47, no. 5, pp. 357–369, 1996.

[10] J. Li and M. Sun, "Scalable term selection fro text categorization," in *EMNLP '07*, 2007, pp. 774–782.

[11] Y. Yang and J. O. Pedersen, "A comparative study on feature selection in text categorization," in *ICML '97*, 1997, pp. 412–420.

[12] M. A. Hasan, S. Salem, B. Pupacdi, and M. J. Zaki, "Clustering with lower bound on similarity," in *PAKDD'09*, 2009, pp. 122–133.

[13] P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in *STOC '98*, 1998, pp. 604–613.

[14] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *SCG'04*, 2004, pp. 253–262.

[15] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, "Arnetminer: Extraction and mining of academic social networks," in *KDD'08*, 2008, pp. 990–998.

[16] Y. Yang, "Noise reduction in a statistical approach to text categorization," in *SIGIR '95*. New York, NY, USA: ACM, 1995, pp. 256–263.

[17] H. Zou, T. Hastie, and R. Tibshirani, "Sparse principal component analysis," *Journal of Computational and Graphical Statistics*.

[18] A. d'Aspremont, F. Bach, and L. E. Ghaoui, "Optimal solutions for sparse principal component analysis," *J. Mach. Learn. Res.*, vol. 9, pp. 1269–1294, 2008.

[19] L. Azzopardi, M. Girolami, and K. van Risjbergen, "Investigating the relationship between language model perplexity and ir precision-recall measures," in *SIGIR '03*, 2003, pp. 369–370.

[20] Y. Yang, N. Bansal, W. Dakka, P. Ipeirotis, N. Koudas, and D. Papadias, "Query by document," in *WSDM'09*, 2009, pp. 34–43.

[21] C. Zhai and J. Lafferty, "A study of smoothing methods for language models applied to ad hoc information retrieval," in *SIGIR'01*, 2001, pp. 334–342.

[22] J. Tang, R. Jin, and J. Zhang, "A topic modeling approach and its integration into the random walk framework for academic search," in *ICDM'08*, 2008.

[23] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, pp. 1157–1182, 2003.

[24] D. D. Lewis and M. Ringuette, "A comparison of two learning algorithms for text categorization," in *SDAIR '04*, 1994, pp. 81–93.

[25] E. D. Wiener, J. O. Pedersen, and A. S. Weigend, "A neural network approach to topic spotting," in *SDAIR '95*, 1995, pp. 317–332.

[26] G. Forman, "An extensive empirical study of feature selection metrics for text classification," *J. Mach. Learn. Res.*, vol. 3, pp. 1289–1305, 2003.

[27] A. Globerson and N. Tishby, "Most informative dimension reduction," in *AAAI '02*, 2002, pp. 1024–1029.

[28] S. Roweis, "Em algorithms for pca and spca," in *NIPS '97*. Cambridge, MA, USA: MIT Press, 1998, pp. 626–632.

[29] E. Bingham and H. Mannila, "Random projection in dimensionality reduction: applications to image and text data," in *KDD'01*, 2001, pp. 245–250.

[30] E. Gabrilovich and S. Markovitch, "Text categorization with many redundant features: using aggressive feature selection to make svms competitive with c4.5," in *ICML'04*, 2004, p. 41.

[31] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *ECML'98*, 1998, ch. 19, pp. 137–142.

[32] M. S. Charikar, "Similarity estimation techniques from rounding algorithms," in *STOC'02*, 2002, pp. 380–388.

[33] A. Andoni and P. Indyk, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," *Commun. ACM*, vol. 51, no. 1, pp. 117–122, January 2008.

[34] T. H. Haveliwala, A. Gionis, and P. Indyk, "Scalable techniques for clustering the web," in *WebDB '00*, 2000, pp. 129–134.

[35] D. Ravichandran, P. Pantel, and E. Hovy, "Randomized algorithms and nlp: using locality sensitive hash function for high speed noun clustering," in *ACL '05*, 2005, pp. 622–629.

[36] D. S. Johnson, "The np-completeness column: An ongoing guide," *J. Algorithms*, vol. 3, no. 2, pp. 182–195, 1982.

APPENDIX

A. Proof of Lemma 3.1

Proof: Following the property of identity for a metric and pseudometric, the proof of the sufficiency is straightforward, and the property of identify for a metric additionally guarantees the necessity if Euclidean distance is chosen. Hence we focus on the proof of the necessity if cosine similarity is chosen.

First of all, we assume that there exist at least two terms w_i and w_j in s with different vector representations $\mathbf{w_i}$ and $\mathbf{w_j}$ respectively, and we denote $\bar{\mathbf{s}} = \mathbf{s}/|s|$, which should be different from either $\mathbf{w_i}$ or $\mathbf{w_j}$. We assume $\mathbf{w_i} \neq \bar{\mathbf{s}}$ without loss of generality, and prove that there exists a document d such that $\texttt{error}_D(d) > 0$, regardless of $\texttt{error}_V$. Since $\mathbf{w_i} \neq \mathbf{w_j}$, there is at least one dimensionality k such that $\mathbf{w_i}_k \neq \mathbf{w_j}_k$ or equivalently $\mathbf{d_k}_i \neq \mathbf{d_k}_j$. When two terms are grouped, the k-th components of $\mathbf{w_i}$ and $\mathbf{w_j}$ are both replaced by $\mathbf{w'}_k$. For document d'_k, we have $\mathbf{d'_k}_i = \mathbf{d'_k}_j$. Therefore, $\frac{\mathbf{d_k}_i}{\mathbf{d_k}_j} \neq \frac{\mathbf{d'_k}_i}{\mathbf{d'_k}_j}$, and subsequently $\frac{\mathbf{d_k}}{||\mathbf{d_k}||} \neq \frac{\mathbf{d'_k}}{||\mathbf{d'_k}||}$. It follows that $\texttt{error}_D(d) = d(\mathbf{d_k}, \mathbf{d'_k}) = d|_{S^{n-1}}(u(\mathbf{d_k}), u(\mathbf{d'_k})) \neq 0$, which completes the proof. ■

B. Proof of Theorem 4.1

Proof: We prove the theorem by a reduction to the GEOMETRIC COVERING BY DISCS problem, a well-known NP-Complete problem [36]. An instance of the GEOMETRIC COVERING BY DISCS problem consists of a set P of integer-coordinate points $\{(p_{i1}, p_{i2})\}_{i=1}^{|P|}$ in the plane, positive integers d and K, and its objective is to determine whether the points of P can be covered by K discs of centroids $\{(c_{j1}, c_{j2})\}_{j=1}^{K}$ and diameter d. Without loss of generality, we may assume $p_{i1}, p_{i2} \geq 1$ for all i, (if not, a translation can be performed on all points of P). And now we construct an instance of TERM FILTERING WITH BOUNDED ERROR problem naturally as follows:

For each point $\mathbf{p_i} = (p_{i1}, p_{i2})$ in P, we add a corresponding term w_i into the vocabulary V, and we create two documents d_1 and d_2 with $\mathbf{d_1} = \{n(w_i, d_1)\}_i = \{p_{i1}\}_i$ and $\mathbf{d_2} = \{n(w_i, d_2)\}_i = \{p_{i2}\}_i$, which means both documents contain the same terms (according to the assumption that the coordinates are all positive integers), but different frequencies of occurrence. We set the error bound for terms $\epsilon_V = d$, and set the error bound for documents $\epsilon_D = +\infty$. It is obvious that if the set of points P are covered by K discs of centroids $\{(c_{j1}, c_{j2})\}_{j=1}^{K}$ and diameter d, then the vocabulary V can be as well partitioned into k subsets or superterms, and vise versa. Therefore, the Term Filtering with Bounded Error problem is NP-hard. ■

The above proof is based on the Euclidean distance metric. For cosine similarity, we can achieve a similar NP-hardness result. Since all samples points are located within an area on the $(n-1)$-dimensional sphere S^{n-1}, $d_C|_{\mathbb{R}^n_{\mathbf{I}} \cap S^{n-1}}$ is isometric to $d_E|_{\mathbb{R}^{n-1}_{\mathbf{I}}}$, the Euclidean distance defined on an area of $(n-1)$-dimensional vector space, and then a mapping function f could be constructed for the proof of NP-hardness. In this sense, the number of documents is required to satisfy $|D| \geq 3$.