Text categorization using topic model and ontology networks

Yinghao Huang, Xipeng Wang and Yi Lu Murphey, Member, IEEE

Abstract—Text categorization based on pre-defined document categories is one of the most crucial tasks in text mining applications in recent decades. Successful text categorization highly relies on the text representations generated from documents. In this paper, an innovative text categorization model, VSM_WN_TM, is presented. VSM_WN_TM is a special Vector Space Model (VSM) that incorporates word frequencies, ontology networks and latent semantic information. Unlike the traditional text representation using only Bag-of-words (BOW) features, it incorporates semantic and syntactic relationship among words such as synonymy, co-occurrence and context, with the purpose of providing more inclusive and accurate text representation. Support Vector Machine is used as document classifier, and the proposed system is evaluated on three publicly available datasets and one domain-specific dataset. Experiment result shows that our approach significantly improves text classification by outperforming approaches such as using only latent features and traditional VSM approaches.

Keywords—Text Categorization, vector space model, ontology network, topic model, support vector machine.

I. INTRODUCTION

As the exponentially increasing amount of digital text in recent decades, text categorization has become one of the most important tasks in text mining applications that have been drawing attention for quite a long time. Especially in the age of “Big Data”, daily load of text document processing brings more and more necessity for accurate and efficient text categorization.

Automatic text categorization process, generally speaking, is the task of building a classifier that assign a pre-defined category to each unknown document. More formally, as described in [1], if $D$ is a set of documents, $D = \{d_1, d_2, ..., d_N\}$, and $C$ is a set of predefined categories, $C = \{c_1, c_2, ..., c_M\}$, the task is to approximate the classifier that maps each $d_i \in D$ to a $c_j \in C$, so that the estimated target mapping function $\hat{\phi} : D \rightarrow C$ coincide the real mapping function $\phi : D \rightarrow C$ as much as possible.

Document category is usually defined based on various application requirements, such as the topic a news article discusses, the importance of a vehicle diagnostic record that describes vehicle repair details [3], the fact stated in a medical diagnostic document that whether or not an injury condition sustains [2], etc. As a result, supervised machine learning algorithms are often applied to learn underlying patterns that connect documents to their assigned categories, so that the classification model can be trained and provides category prediction for unknown documents.

There are already plenty of research works that focus on developing and improving machine learning techniques adopted for building classification models, such as using k-nearest-neighbor classifiers [8,12], Naïve Bayes classifiers [5,11], Self Organizing Maps [2,10], Neural Network classifiers [6,9,13] and Support Vector Machine classifiers [4,7]. By using these techniques, promising text classification performance could be obtained. However, whatever machine learning techniques are used to build these classifiers, the classification accuracy will be “bottlenecked” if the representation quality of the document is poor, i.e., the representation of a document does not reflect close relationship with its assigned category. For example, the typical and mostly used approach of representing a text document is the Vector Space Model (VSM) [14], in which each document is represented by a weighted vector that provide a more convenient way for computation and analysis. However, this approach does not consider the semantic relationship between words, such as synonyms, hyponyms, etc. The classification accuracy is especially deteriorated in the document set that different category has similar word occurrence [2,15]. As a result, a text categorization model that captures underlying semantic and syntactic information besides single word occurrence features is of great necessity.

Considering the above fact, researchers have been working on improving document representations for classification, by altering the VSM model. Some typical approach include applying different weighting scheme to each term [25], adding semantic information as additional features, by capturing word co-occurrence, such as [16], or building word ontology as external knowledge that provides synonym information, such as [17,18,19]. Note here, word ontology is defined as specification of a representational vocabulary for a shared domain of discourse that describes word relationships, according to [20]. However, this requires a lot of manual or semi-manual work to build word ontology from scratch, and this ontology is usually limited by domain-specific corpus. Therefore, to a large extent, current research work mainly focuses on learning ontologies from existing resources such as English lexicons [24]. Furthermore, most of the state-of-art approaches use only nouns for ontology building, and a large extent of methods aims at constructing IS-A-related concept hierarchies. [21-23].
The above approaches, in fact, still only consider relationship between words instead of underlying semantic meanings. More advanced research about digging out latent semantic components could be referred to topic modeling, which is typically used for unsupervised text clustering, information retrieval and dimension reduction, such as Latent Semantic Indexing (LSI) [26], probabilistic latent semantic analysis (PLSA) [27], and Latent Dirichlet Allocation (LDA) [28]. These language models aim at estimating latent semantic components that are mapped from the word space. However, most of the work been done on applying topic model to text categorization is to purely use estimated latent features for classification, such as [15,29,30,31], which is claimed in [32] to be less accurate than using bag-of-word (BOW) features, especially when training data size is large.

In this paper, we present an inventive text categorization approach that focuses on augmenting VSM model with both ontology network and topic model. We first modify the original BOW representation by using an innovative global weight scheme, and then combine the weighted BOW matrix with a synset matrix based on WordNet ontology, and also with a latent topic matrix generated from PLSA model. Our system is evaluated using SVM classifier, on three publicly available datasets and a domain-specific dataset.

The remainder of the paper is organized as follows: Section 2 details our text categorization system, Section 3 presents the case study and discusses the empirical results, and Section 4 concludes the paper.

II. METHODOLOGY

In this section, we present the proposed text categorization system and algorithms in each system component, following the framework shown in Figure 1. Our system takes in the training document collection and generates a list of indexed terms. After that, each indexed terms are weighted, and the document corpus is modeled by traditional VSM as a weighted TDW matrix. PLSA model is applied to generate a “latent topic” level (LTD) matrix, and WordNet ontology is feed into the system to generate a new term-document matrix, and a “synset” level (SD) matrix. These matrices are then combined together for final document representation, and used for SVM classifier training. These matrices are then combined together for final document representation, and used for SVM classifier training. More details will be discussed in the following sub-sections.

A. Build Vector Space Model

As discussed in Section 1, text document is usually represented by VSM for the ease of computation and analysis. A vector space model should be built based on carefully selected terms and weighting schemes [35]. The vector space model generation consists of two stages: word indexing and term weighting scheme. First of all, the documents are represented as a set of keywords called indexed terms. After document indexing, a proper global weighting scheme is selected and applied to the vectors to adjust the influence each term may have on the model based on their appearing frequency.

1) Generating VSM model for a document

For a given set of training documents Tr, 
\[ Tr = \{D_1, D_2, ..., D_N\} = T_1 \cup T_2 \cup ... \cup T_C \], where \( D_i \) is the \( i^{th} \) training document, \( C \) is the number of document categories, and \( T_c \) is the set of documents that belong to category \( c \), \( c = 1,2,\ldots,C \), our vector space model is built through the following machine learning process:

![Figure 1. Proposed text categorization system framework](image-url)
text categorization [35]. Global weighting schemes should be applied to each indexed term with the purpose of reducing or enhancing the effect they have on particular document. A number of term weighting schemes can be found in [36]. Examples of a well-known global weight schemes used in text mining is inverse document frequency (idf), which is defined as:

\[ idf_i = \log_2 \left( \frac{N - \text{df}_i}{\text{tf}_i} \right) + 1, \]

where \( \text{tf}_i \) is the occurrence frequency of term \( t_i \) within document \( D_j \); \( \text{df}_i \) is the document frequency, i.e., the total number of documents in the document collection that contain \( t_i \), and \( N_d \) is the total number of documents in the training data set. When idf is used as the global weight function, we have \( g_i = \text{idf}_i \).

2) An innovative global weighting scheme

Although there are plenty of global weight approaches available, most of them are designed for the entire dataset, i.e., training document corpus \( Tr \). Based on our observation, important term words or their synonyms appear frequently in documents in a specific category, especially when the user defined category is determined by some specific keywords [2]. As a result, we developed the following category-entropy weighting function, denoted as \( CE_W \):

1. For each term \( t_i \) in the term list \( T_L \), calculate the proportion of the documents in \( Tr \) that contain \( t_i \) within \( C \) different categories.

\[ p_{ij} = \frac{N \cdot c_{ij}}{N \cdot c_j}, j = 1,2,...,C, \]

where \( N \cdot c_{ij} \) is the number of documents within the \( j^{th} \) categories that contains \( t_i \), and \( N \cdot c_j \) is the total number of documents in the \( j^{th} \) category.

2. Normalize \( p_{ij} \), so that \( \alpha_{ij} = \frac{p_{ij}}{\sum_{j=1}^{C} p_{ij}} \).

3. Calculate the entropy with respect to \( t_i \):

\[ E_i = \sum_{j=1}^{C} -\alpha_{ij} \log \alpha_{ij}. \]

The entropy measure is a good indicator of how term \( t_i \) is distributed over different document categories.

The higher the entropy, the less important item \( t_i \) is, since it is more evenly distributed among the document categories.

4. Calculate the global weight \( CE_W_i \) for \( t_i \):

\[ CE_W_i = 1 - \frac{E_i}{\log C}, \]

where \( C \) is the total number of categories. This global weight function gives more weights to terms that have small entropy values.

We will show in Section 3 that the category-entropy based global weight function performs better than the inverse document frequency (idf) method.

At the end of VSM generation step, the output is a TDW matrix \( M_0 = [W_{1j}^T, W_{2j}^T, ..., W_{nj}^T] \).

B. A VSM augmented with ontology network

External or background knowledge has been found useful in improving text categorization, especially for short or ambiguous documents [37, 38]. It has a great advantage in helping extract semantic relationships, match important phrases, strengthen co-occurrences, etc. As discussed in Section 1, WordNet is one of the best known sources of external knowledge used for text categorization. It contains a network of semantically related words. Words are grouped into semantic groups (synsets) that provide word synonyms together with short explanations and general definitions. For the purposes of text categorization, it is mostly used to unify the vocabulary across the documents by modifying the document features with use of the related words [24].

In our proposed text categorization system, WordNet is used in two ways, derived from and modified based on basic approaches introduced in [21]: “add” and “repl” rules.

For each indexed term word \( t_i \) generated by VSM model, we can use POS tagging such as Stanford POS tagger [39] to identify its lexical category, and then find list of synonyms \( S_i \) for \( t_i \) in WordNet ontology. However, in a lot of applications, POS tagging may not be very reliable, e.g., text documents are noisy and lack of grammar structure and sentence boundary [3]. As a result, we generate the synset for \( t_i \) only considering one word class from “Noun”, “Verb” or “Adjective”, and choose the best one based on categorization performance, which will be discussed in section III.

1) Update term-document matrix (modified “repl” rule)

Under this rule, the term-document matrix \( M_0 \) generated in II-A is updated using WordNet, in a way that for a term \( t_i \) that has synset \( S_i \), its weight in document \( D_j \) is updated using the following equation:

\[ w_{ij} = \max(\forall t_r \in S_i, t_r \in T_L). \]

The above equation ensures that similar terms share the same weighting value, so that they are considered as equally important. For example, if term \( t_{xy} \) = “entire” appears in document \( A \) and \( t_y \) = “total” appears in document \( B \), and suppose \( S_x = S_y = \{ t_x, t_y \} \), then we will have \( w_{xA} = w_{yA} = w_{xB} = w_{yB} \). The output from this stage is an updated TDW matrix \( M_1 \) that has the same dimension as \( M_0 \).

2) Generate synset -document (SD) matrix (“add” rule)
Under this rule, a SD matrix $M_c$ is generated using WordNet by introducing the “synset” level features, which represents the group of synonyms for each term. Mathematically, for a document $D_i \in Tr$, its “synset” vector representation $Q_{D_i}$ is defined as following:

$$Q_{D_i} = [q_{i1}, q_{i2}, \ldots, q_{iV}],$$

where $q_{il}$ denotes the summation of term weighting values within each group of synonyms, $S_i$ represents the $i^{th}$ synonym group, and $V$ represents the total number of synonym groups generated. Thus, we have:

$$M_c = [Q_{D_1}^T, Q_{D_2}^T, \ldots, Q_{D_N}^T].$$

C. A VSM augmented with PLSA topic modeling

1) Learning PLSA model from training documents

PLSA model is a well-known statistical language model for text clustering and information retrieval [27]. It represents a document with a convex combination over “latent topics”, which are estimated by updating and fitting model parameters in order to maximize the joint probability of observed document-word pairs. It is a statistical variant of LSI [26] and based on a statistical generative model called Aspect Model [27]. The starting point of PLSA is the term-document frequency (TDF) matrix before applying global weight scheme, and it follows the bag-of-words assumption, in which each word appears independently, and the occurring order of each word is not considered. Figure 2 shows the graphical model representation of PLSA, based on Bayesian Networks [40].

![Graphical model representation of PLSA](Image)

Figure 2. Graphical model representation of PLSA

In the above graphical model, the solid circles $D$ and $t$ represents a document and a term that are observed by people. PLSA model is a generative model that assumes there is a latent “topic” variable $z$ between documents and terms. The two rectangles represents number of sample documents or words observed, and $P(D)$, $P(z \mid D)$, $P(t \mid z)$ represents the probabilities of observing a document $D$, a latent topic $z$ occurring in $D$, and word $t$ belonging to $z$, respectively.

The typical approach of PLSA modeling, is to estimate $P(z \mid D)$ and $P(t \mid z)$, for each document-topic and term-topic pair, by maximizing the following log-likelihood function:

$$L = \sum_{D_i} n(D_i, t) \log P(D_i) \sum_z P(t \mid z) P(z \mid D_i), \quad (1)$$

where $n(D_i, t)$ denotes the term frequency of $t$ appears in document $D_i$.

The generation process of PLSA model for training document corpus is proposed as following:

1. Select a document $D$ from $Tr$ based on $P(D)$.
2. Pick a topic $z$ according to $P(z \mid D)$.
3. Given $z$, generate a word $t$ based on $P(t \mid z)$.

The variables $P(z \mid D)$ and $P(t \mid z)$ are what we are interested in and want to estimate. Since $P(D)$ is not related to the parameter we want to estimate and we assume that it is constant among documents in $Tr$, we then have:

$$\text{arg max}(L) \propto \text{arg max} \sum_{D_i} n(D_i, t) \log \sum_z P(t \mid z) P(z \mid D_i) \quad (2)$$

This maximization likelihood estimation can be solved using Expectation Maximization (EM) algorithm [41]. Each iteration of EM algorithm consists of expectation step (E-step) and maximization step (M-step). In E-step, based on the current estimated $P(z \mid D)$ and $P(t \mid z)$, the posterior probability of $P(z \mid D, t)$ is computed for each document-word pair. In M-Step, $P(z \mid D)$ and $P(t \mid z)$ are updated by maximizing equation (2). Detailed steps of EM algorithm are discussed below:

**Initialization:** First determine maximum number of iterations $R$, and number of topics $G$ to be generated. For each document-topic and topic-word pair, assign random values to $P_0(z \mid D)$ and $P_0(t \mid z)$.

**E-step:** At iteration $r$, for each observed topic, word and document, $z^{(r)}$, $t^{(r)}$, and $D^{(a)}$, compute:

$$P_r(z^{(r)} \mid D^{(a)}, t^{(r)}) = \frac{P_r(t^{(r)} \mid z^{(r)}) P_r(z^{(r)} \mid D^{(a)})}{\sum_z P_r(t^{(r)} \mid z) P_r(z \mid D^{(a)})} \quad (3)$$

where $P_r(z^{(r)} \mid D^{(a)})$ and $P_r(t^{(r)} \mid z^{(r)})$ are derived from iteration $r-1$.

**M-step:** At iteration $r$, for each document-topic and topic-word pair, compute $P_{r+1}(z^{(r)} \mid D^{(a)})$ and $P_{r+1}(t^{(r)} \mid z^{(r)})$ based on the following updating formulas:

$$P_{r+1}(t^{(r)} \mid z^{(r)}) = \frac{\sum_{D_i} n(D_i, t^{(r)}) P_{r}(z^{(r)} \mid D_i, t^{(r)})}{\sum_{D_i} n(D_i, t) P_{r}(z^{(r)} \mid D_i, t)} \quad (4)$$

$$P_{r+1}(z^{(r)} \mid D^{(a)}) = \frac{\sum_{D_i} n(D^{(a)}, t) P_{r}(z^{(r)} \mid D^{(a)}, t)}{\sum_{t} n(D^{(a)}, t)} \quad (5)$$
More detailed derivation of (3) to (5) can be referred to [27], [42] and [43].

The output from this stage after EM learning is a latent topic-document (LTD) matrix \( M_{id} \), in which each document has a vector representation \( H_{D_i} \) that is mapped from indexed term space to latent topic space, \( H_{D_i} = [P_{i1}, P_{i2}, ..., P_{iG}] \), and \( P_{id} = P_B(z^{(i)} | D^{(i)}) \), \( i = 1, 2, ..., G \), where \( R \) denotes the maximum number of iterations EM went through, and \( G \) denotes the number of topics generated. As a result, we have:
\[
M_{id} = [H_{D_i}^T, H_{D_2}^T, ..., H_{D_N}^T].
\]

2) Generate topic-document vector for unknown document

Although PLSA is originally designed for unsupervised learning, it can be easily extended to unknown documents. For a testing document \( D_u \), we run through EM algorithm again, with all other parameters kept fixed, except \( P(z | D_u) \). In E-step, based on the current estimated \( P(z | D_u) \), the posterior probability of \( P(z | D_u, t) \) is computed. In M-Step, only \( P(z | D) \) is updated by equation (5). Therefore, a vector representation \( H_{D_u} \) is generated for \( D_u \), with the same dimension as \( H_{D_i} \).

D. Generate hybrid VSM model for classification

The matrixes generated in above sections are combined together as the final VSM representation of documents in \( Tr \). The process of VSM matrix generation and augmentation is shown in the following Figure 3. The combined matrix is then used to learn and evaluate classification model, e.g., SVM, Neural Networks, Naïve Bayes classifier, etc.

![Figure 3. VSM matrix generation and augmentation](image)

III. EMPIRICAL CASE STUDY

In this section, we present experiments we conducted using our proposed text categorization framework, and classification results on several different datasets, in terms of accuracy.

A. Datasets

In this empirical study, we first use three publicly available and widely used datasets to evaluate our proposed system. These datasets include Reuters-21578 [45], Nist Topic Detection and Tracking corpus (TDT2) [44], and 20 newsgroups [46]. Reuters-21578 corpus contains 21578 documents in 135 categories. After removing documents with multiple category labels, it left 8293 documents in 65 categories. In TDT2, those documents appearing in two or more categories were removed, and only the largest 30 categories were kept, thus leaving 9,394 documents in total. 20 newsgroups dataset is a collection of 18846 newsgroup documents, partitioned (nearly) evenly across 20 different newsgroups.

To evaluate the system performance on domain-specific datasets that has customized category definition such as [2], we also used a dataset named VDR, that contains 600 vehicle diagnostic records, in which documents that contain descriptions that reveal systematic engineering or manufacturing failures are defined as of interests (Category-A), and all other documents belong to Category-B. The major challenge in this problem is that the documents of interests are not explicitly defined by either topics or general descriptions, as shown in the following examples:

- **Category-A document:** “perform abs self roadtest found rear wheel speeds sensor connector corroded into sensor replace sensor and connector road tester ok clear code”
- **Category-B document:** “road roadtest traction control lamp on eec roadtest code c1280 u415 om rcm contact hot line 103912699 check connection at rcm check mounting bolts ok clear code”

In all of these datasets discussed above, preprocessing tasks mentioned in Section II-A are conducted and stop words are removed. Note here, all words having occurrence frequency lower than \( \tau = 5 \) are removed, except VDR dataset. TDW matrix weighted by \( CE_W \) discussed in Section II-A is generated for each dataset. TDF matrix is also generated for PLSA model learning, and TDW matrix weighted by \( idf \) is generated for evaluation purpose.

B. Augment VSM with WordNet

WordNet ontology network is utilized in our system not only to update TDW matrix, but also to create new synset features. In our experiments, it generates 2628, 4384, 3063 and 696 synset features for Reuters, TDT2, 20newsgroups and VDR, respectively. In our experiments, we first look into the effect of text categorization using terms within different word class. The results are shown in Table I. It is obvious that the best word class is “Noun”, which only generates 377, 1161, 621 and 21 “synset” features for Reuters, TDT2, 20newsgroups and VDR, respectively, and having a promising text categorization accuracy.
3-fold cross validation, and in each fold, we choose 2/3 documents from each class as training set, and the remaining 1/3 documents as testing set. We apply the Gaussian Radial Basis kernel function (RBF) and tune the parameter gamma to 0.001, 0.001, 0.1 and 0.1, for Reuters, TDT2, 20newsgroups and VDR, respectively, based on the average testing accuracy of the 3 folds.

E. Experiment results & analysis

The classification results are presented in the following Table I. We evaluate several systems as our baseline, including TDW matrix weighted by idf; and using only PLSA generated LTD matrix. From the result, it is obvious that global weighting scheme CE_W outperform the idf weighting, and the SD matrix generated by WordNet improves categorization accuracy by combining with the idf weighted TDW matrix. Furthermore, the final proposed system, with CE_W weighted and WordNet updated TDW matrix, plus SD matrix generated by WordNet and LTD matrix generated by PLSA, significantly outperforms all other systems, indicating that adding both word relationships and latent semantic information could improve text representation.

IV. CONCLUSION

This paper proposes an innovative text categorization model, VSM_WN_TM, based on Vector Space Model (VSM), WordNet ontology, and PLSA topic modeling. Support Vector Machine is used as document classifier, and the proposed system is evaluated on publicly available datasets and domain-specific dataset. Experiment result shows that incorporating semantic and syntactic relationship among words such as synonymy, co-occurrence and context could greatly improve text representation, and our approach significantly outperforms conventional approaches such as using only BOW features or latent topic features.

Table I. Text categorization performance using WordNet based on different word class

<table>
<thead>
<tr>
<th>Categorization accuracy</th>
<th>Noun</th>
<th>Verb</th>
<th>Adjective</th>
<th>Mixed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reuters</td>
<td>92.74%</td>
<td>92.34%</td>
<td>89.01%</td>
<td>92.26%</td>
</tr>
<tr>
<td>TDT2</td>
<td>97.07%</td>
<td>96.22%</td>
<td>96.15%</td>
<td>96.64%</td>
</tr>
<tr>
<td>20news</td>
<td>87.68%</td>
<td>86.48%</td>
<td>86.58%</td>
<td>86.73%</td>
</tr>
<tr>
<td>VDR</td>
<td>84.53%</td>
<td>82.32%</td>
<td>82.32%</td>
<td>83.97%</td>
</tr>
</tbody>
</table>

Table II. Text categorization system accuracy comparison

<table>
<thead>
<tr>
<th></th>
<th>Reuters</th>
<th>TDT2</th>
<th>20news</th>
<th>VDR</th>
</tr>
</thead>
<tbody>
<tr>
<td>TDW matrix</td>
<td>91.03%</td>
<td>89.37%</td>
<td>85.85%</td>
<td>80.95%</td>
</tr>
<tr>
<td>weighted by idf</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TDW matrix</td>
<td>92.10%</td>
<td>96.01%</td>
<td>87.25%</td>
<td>85.18%</td>
</tr>
<tr>
<td>weighted by CE_W</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TDW matrix</td>
<td>91.42%</td>
<td>95.09%</td>
<td>87.10%</td>
<td>83.07%</td>
</tr>
<tr>
<td>weighted by idf + SD</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>matrix</td>
<td>84.60%</td>
<td>90.24%</td>
<td>79.46%</td>
<td>82.32%</td>
</tr>
<tr>
<td>LTD matrix by PLSA</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Updated TDW matrix</td>
<td>93.06%</td>
<td>98.78%</td>
<td>88.84%</td>
<td>87.84%</td>
</tr>
<tr>
<td>weighted by CE_W + SD</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>matrix + LTD matrix</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

C. Augment VSM with PLSA

With the purpose of being consistent, in PLSA learning, for all datasets, we define the maximum number of iterations \( R = 500 \) for training set, and \( R = 200 \) for testing set. Number of latent topics is defined as \( G = 40 \). These parameters can be tuned for optimized results, which will also be investigated in the next step of work. Also, the convergence goal is defined as \( \epsilon = 1E^{-5} \). An example of log-likelihood function maximization on Reuters dataset is shown in Figure 4. It is obvious that the log-likelihood function converges very fast and become very stable after 500 iterations.

D. SVM training

Considering that the focus of this work is not improving or comparing machine learning algorithms, we use SVM as our classification model throughout different experiments. SVM training is carried out with LIBSVM package, which is developed by Chih-Chung Chang and Chih-Jen Lin from National Taiwan University [47]. For each dataset, we did 500 iterations.


[36] Liping Huang, Yi Lu Murphy: Text Mining with Application to Engineering Diagnostics. *IEA/AIE 2006*: 1309-1317


[44] Deng Cai, Xiaofei He and Jiawei Han, "Document Clustering Using Locality Preserving Indexing", IEEE TKDE 2005.

[45] Reuters-21578, Distribution 1.0. Web document. URL: http://www2.dalleidewillis.com/resources/testcollections/reuters21578
