Applied Intelligence (2020) 50:2339–2351
https://doi.org/10.1007/s10489-020-01680-w

A multi-label text classification method via dynamic semantic representation model and deep neural network

Tianshi Wang[1] · Li Liu[1] · Naiwen Liu[1] · Huaxiang Zhang[1] · Long Zhang[1] · Shanshan Feng[1]

Published online: 5 March 2020
© Springer Science+Business Media, LLC, part of Springer Nature 2020

Abstract

The increment of new words and text categories requires more accurate and robust classification methods. In this paper, we propose a novel multi-label text classification method that combines dynamic semantic representation model and deep neural network (DSRM-DNN). DSRM-DNN first utilizes word embedding model and clustering algorithm to select semantic words. Then the selected words are designated as the elements of DSRM-DNN and quantified by the weighted combination of word attributes. Finally, we construct a text classifier by combining deep belief network and back-propagation neural network. During the classification process, the low-frequency words and new words are re-expressed by the existing semantic words under sparse constraint. We evaluate the performance of DSRM-DNN on RCV1-v2, Reuters-21578, EUR-Lex, and Bookmarks. Experimental results show that our method outperforms the state-of-the-art methods.

Keywords Text classification · Word embedding · Clustering · Sparse representation · Neural network

1 Introduction

The development of computer technology and the explosive growth of text data cause the increase of text processing workload [8], and promote the research of many subfields in natural language processing (NLP). Text classification, which refers to dividing the texts into different categories according to its topic, content, and attributes, has gradually become a hot issue. Furthermore, multi-label text classification is to assign one or more category labels to each text, which is widely used in sentiment analysis, information retrieval [9], news subject classification, and spam detection [20].

However, there are still many problems to be solved in multi-label text classification. For example, the appearance of new words reduces the accuracy of the classifier, a great number of textual data affect the convergence of neural networks and some semantic information is ignored during the text quantization. Besides, due to the rapid growth of unstructured texts from social networks, mail systems and other platforms, text mining as the basis of text classification has also attracted great attention from researchers.

To address the above issues, we propose a novel text classification method that combines dynamic semantic representation model and deep neural network classifier (DSRM-DNN). The proposed method improves the classification performance by updating text mining technology and optimizing text classifiers. In summary, the contributions of the paper are as follows:

- Dynamic semantic representation model is proposed to quantify the texts. The model combines word embedding model and clustering algorithm to select semantic words, then utilizes bag of words (BOW) and word attributes to obtain textual features. Dynamic semantic representation not only greatly reduces the complexity of the existing BOW, but also lays the foundation for the expression of new words.
- A sparse constraint is adopted to build the relationship between new words and the existing semantic words. By utilizing the semantic words in DSRM to obtain the sparse representation of low-frequency words and new words, more words can be quantified and used in the training data.
- An efficient multi-label text classifier is constructed based on deep belief network (DBN) and BP network. DBN is

✉ Li Liu
liuli_790209@163.com

Tianshi Wang
1056164931@qq.com

[1] School of Information Science and Engineering, Shandong Normal University, Jinan, China

used to preprocess the input of the BP network, which reduces the risk of falling into local optimum and accelerates the convergence of the classifier network.

The rest of the paper is organized as follows. Section 2 introduces the related work of multi-label text classification. In Section 3, the details of the proposed method are described. Experimental results and discussions are presented in Section 4. Finally, Section 5 summarizes work and gives further research direction.

2 Related work

Compared with simple classification tasks, multi-label text classification has two characteristics. On one hand, the corpus cannot be directly input into the classifier, and the texts must be quantified for further purposes. On the other hand, different from the multi-class classifier, each sample in multi-label data belongs to several target domains that are unrelated to each other. The final performance of multi-label text classification is jointly affected by the text quantization methods and multi-label text classifiers.

2.1 Text quantization

The text quantization methods [5, 6, 13, 15] can be roughly divided into three categories: the traditional language models, the sequence or structured language models, and the attention language models. The traditional language models based on BOW [14] are widely used in the data pre-processing of various research fields, such as text classification, cross-modal retrieval [10, 23], and so on. The second kind of language models can use sequence or structured models to quantify texts, mainly including convolutional neural network, recurrent neural network, recursive auto-encoders, etc. The third kind of language models is based on attention mechanisms to build quantification features by scoring the words or sentences differentially. The attention mechanisms are commonly known as hierarchical attention and self-attention.

In the traditional representation models, textual data are usually unordered word sets that ignore grammar and the order of words, but they are widely used in NLP because of their simplicity and efficiency. To reduce the resource consumption, some representative words in the corpus are selected as the basic elements of BOW. Statistical methods and theme models [33] are commonly used for keyword selection. The statistical method mainly extracts keywords according to the word attributes (i.e. term frequency, TF-IDF, etc.), and the theme model is based on the following two assumptions: each document is a mixed distribution of several topics and each topic follows the probability distribution of words. At present, the LDA theme model that combines PLSA model and Dirichlet prior distribution is widely used. In addition to the above two methods, the methods based on graph theory [32] and complex networks can also be used to extract keywords.

2.2 Multi-label classifier

With the development of neural network research, deep learning has been extensively studied and widely used in many areas. The construction of text classifiers can be divided into two categories: one is based on the traditional machine learning algorithm, the other is based on deep neural network.

Before the emergence of deep neural networks, most of the studies were based on traditional machine learning algorithms, namely the artificial definition of machine learning algorithms and the construction of experimental models. At present, a large number of machine learning methods are applied in the text categorization system, such as Naive Bayes [26] based on Bayesian theory and characteristic conditional independence hypothesis, the support vector machine (SVM) based on the statistical learning VC theory and the structural risk minimum principle, and the decision tree method [4] of the optimal scheme which is obtained by comparing different schemes with probability and tree in graph theory. In addition, there are also k-nearest neighbor classification models [3], classification methods based on association rules and so on [11, 21, 22]. Although time complexity of the above models is low, the context and potential semantic relationship of text words are not fully considered.

Deep neural networks [1] can mine more complex text semantics and provide a new research direction for NLP. At present, convolutional neural network [7, 12], back-propagation network and recurrent neural network have been applied in text classification and achieved good experimental results.

3 The proposed method

In this section, we introduce the overall process of DSRM-DNN shown in Fig. 1. The proposed method consists of the text quantization module and the classifier module. The text quantization module is divided into three subtasks: semantic word extraction, word feature construction, and dynamic representation. First, DSRM-DNN selects semantic words by combining word embedding model and clustering algorithm. Then the selected words are taken as the elements of DSRM-DNN and quantified by the weighted combination of word attributes. During the classification process, DSRM-DNN can represent the low-frequency words and new words according to the existing words in DSRM-DNN through sparse representation. In the classifier module, deep belief network and back-propagation neural network are utilized to construct a text classifier.

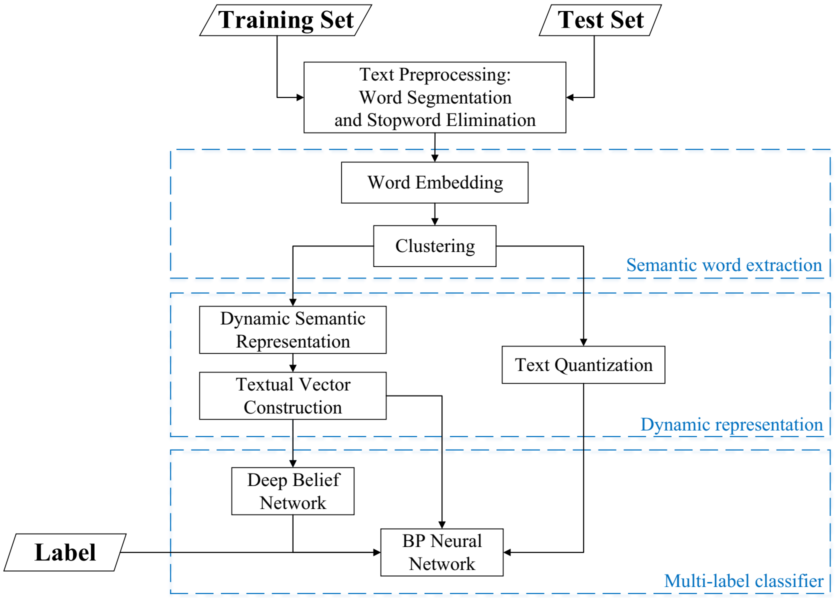
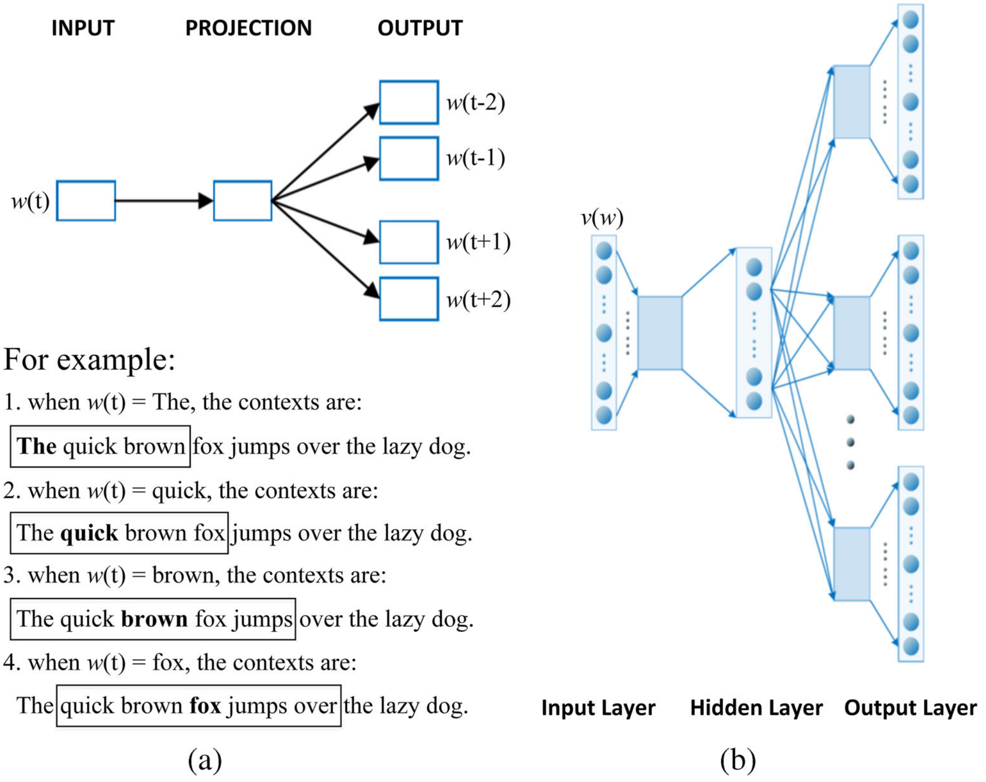


Fig. 1 The framework of the text classification

3.1 Semantic word extraction

Before extracting semantic words, we preprocess the corpus $\widetilde{D}$~, including uniform format, word segmentation and stopword elimination. Therefore, we can obtain the preprocessed corpus D and the word sets T_j corresponding to the j th text.

During the words extraction, word embedding model is trained on the word sets. Specifically, the skip-gram model based on hierarchical softmax is adopted to obtain the embedding vectors. Thus, the text semantics are quantized and stored in the word embedding vectors. In the word embedding model, we can regard each leaf node of the binary tree as a word, and each non-leaf node is equivalent to a perceptron that outputs 0 or 1.



For example:

1. when w(t) = The, the contexts are:

The quick brown fox jumps over the lazy dog.

2. when w(t) = quick, the contexts are:

The **quick** brown fox jumps over the lazy dog.

3. when w(t) = brown, the contexts are:

The quick **brown** fox jumps over the lazy dog.

4. when w(t) = fox, the contexts are:

The quick brown **fox** jumps over the lazy dog.

Fig. 2 **a** is the skim-gram model concept diagram with window size 5, w(t) is a word in the current position t in the text, and other marks are defined in the same way. **b** is the skim-gram network structure diagram, $v(w)$ is the vector representation of w

Each word in the datasets can be represented by a unique encoding, and its encoding sequence corresponds to the sequence of events. Skip-gram is a language model that predicts the context according to the current words, and Fig. 2 shows the model. For ease of reference, Table 1 lists the major notations used in the embedding model and their mathematical meanings.

The learning objective of the embedding model can be written as the maximum likelihood function:

$$\mathcal{L} = \sum_{w \in c} logp(Context(w)|w) \tag{1}$$

The probability function of the language model is

$$p(Context(w)|w) = \prod_{u \in Context(w)} p(u|w) \tag{2}$$

where w is the word in the corpus, u is the word in the context of w. Each u is independent of each other in the Hierarchical Softmax, so we get:

$$p(u|w) = \prod_{j=2}^{l^u} p\left(d_j^u | v(w), \theta_{j-1}^u\right) \tag{3}$$

where each multiplier in (3) is logistic regression:

$$p\left(d_j^u | v(w), \theta_{j-1}^u\right) = \begin{cases} \sigma\left(v(w), \theta_{j-1}^u\right), & d_j^u = 0 \\ 1-\sigma\left(v(w), \theta_{j-1}^u\right), & d_j^u = 1 \end{cases} \tag{4}$$

Given that d is 0 or 1, we can easily express (4) in exponential form:

$$p\left(d_j^u | v(w), \theta_{j-1}^u\right) = \left[\sigma\left(v(w), \theta_{j-1}^u\right)\right]^{1-d_j^u} \cdot \left[1-\sigma\left(v(w), \theta_{j-1}^u\right)\right]^{d_j^u} \tag{5}$$

Substituting (2), (3), (5) into (1), we get the new objective function:

$$\mathcal{L} = \sum_{w \in u} \sum_{u \in Context(w)} \sum_{j=2}^{l^u} \left(1-d_j^u\right) \cdot log\left[\sigma\left(v(w), \theta_{j-1}^u\right)\right] + d_j^u \cdot log\left[1-\sigma\left(v(w), \theta_{j-1}^u\right)\right] \tag{6}$$

Table 1 Summary of symbols in the word embedding model

Symbols	Meanings
p^w	the path from the root node to the leaf node
l^w	the number of nodes in the path p^w
$p_1^w, p_2^w, \cdots, p_{l^w}^w$	each node in the path p^w
$d_2^w, d_3^w, \cdots, d_{l^w}^w$	d_j^w is the code of the j th node in the path p^w
$\theta_1^w, \theta_2^w, \cdots, \theta_{l^w-1}^w$	parameter vector corresponding to non-leaf nodes in path p^w

Each item in the objective function is denoted as:

$$\begin{aligned} \mathcal{L}(w,u,j) = & \left(1-d_j^u\right) \cdot log\left[\sigma\left(v(w), \theta_{j-1}^u\right)\right] \\ & + d_j^u \cdot log\left[1-\sigma\left(v(w), \theta_{j-1}^u\right)\right] \end{aligned} \tag{7}$$

u is certain for a given training instance w and its context $\{u \in c\}$, so there are only two variables $v(w)$ and θ_{j-1}^u in (7).

The partial derivatives of $\mathcal{L}(w,u,j)$ with respect to $v(w)$ and θ_{j-1}^u are given as follows:

$$\frac{\partial \mathcal{L}(w,u,j)}{\partial \theta_{j-1}^u} = \left[1-d_j^u-\sigma\left(v(w), \theta_{j-1}^u\right)\right] v(w) \tag{8}$$

$$\frac{\partial \mathcal{L}(w,u,j)}{\partial v(w)} = \left[1-d_j^u-\sigma\left(v(w), \theta_{j-1}^u\right)\right] \theta_{j-1}^u \tag{9}$$

Therefore, we get the update functions for $v(w)$ and θ_{j-1}^u:

$$\theta_{j-1}^u = \theta_{j-1}^u + \eta\left[1-d_j^u-\sigma\left(v(w), \theta_{j-1}^u\right)\right] v(w) \tag{10}$$

$$v(w) = v(w) + \eta \sum_{u \in Context(w)} \sum_{j=2}^{l^u} \frac{\partial \mathcal{L}(w,u,j)}{\partial v(w)} \tag{11}$$

where η is the learning rate.

Algorithm 1 Word embedding modal.

```
Input: word data $\{T_i\}$, word embedding dimension $e$,
       context window size $c$.
Output: word embedding $v(w)$.
randomly initialize $v(w)$, and initialize $s$=0.
for $u \in context(w)$ do
    for $j = 2$ to $l^u$ do
        $q = \sigma(v(w), \theta_{j-1}^u)$;
        $g = \eta(1 - d_j^u - q)$;
        $s = s + g\theta_{j-1}^u$;
        $\theta_{j-1}^u = \theta_{j-1}^u + gv(w)$.
    end
end
$v(w) = v(w) + s$.
```

After several update iterations, we can get the embedding vector $v(w)$ of each word and the feature matrix T_i, where each column in the feature matrix is the feature representation of a word. Then clustering analysis [16–18] is carried out for each text, and the words with similar semantics are divided into the same class. During each clustering process, the clustering center is extracted as the semantic words of the text.

Assuming that the set of embedding vectors of a text is $\{w_1,w_2,\cdots,w_n\}$, **S** is defined as the similarity matrix between samples, and $s(i,j) > s(i,k)$ if and only if the similarity between w_i and w_j is greater than that between w_i and w_k. In addition, we defined **R** as the responsibility matrix and **A** as the availability matrix, where $r(i,k)$ describes the suitability of w_k for the clustering center of w_i and $a(i,k)$ describes the suitability of w_i to select w_k as its clustering center, and the elements of matrices **R** and **A** are initialized to 0.

The clustering algorithm is implemented by iteratively updating the responsibility $r(i,k)$ and availability $a(i,k)$:

$$\mathbf{R}_{t+1}(i,k) = (1 - \lambda)\mathbf{R}_{t+1}(i,k) + \lambda\mathbf{R}_t(i,k) \tag{12}$$

where $\mathbf{R}_{t+1}(i,k) = \left\{ \mathbf{S}(i,k) - \max\limits_{j\neq k} \mathbf{A}_t(i,k) + \mathbf{R}_t(i,k), i\neq k\mathbf{S} \right.$ $(i,k) - \max\limits_{j\neq k}\mathbf{S}(i,k), i = k$, λ is the damping coefficient.

$$\mathbf{A}_{t+1}(i,k) = (1 - \lambda)\mathbf{A}_{t+1}(i,k) + \lambda\mathbf{A}_t(i,k) \tag{13}$$

where $\mathbf{A}_{t+1}(i,k) = \{\min 0,\ \mathbf{R}_{t+1}(k,k) + \sum_{j\neq i,k}\max$ $\mathbf{0}, \mathbf{R}_{t+1}(j,k), i\neq k \sum_{j\neq k}\max\mathbf{0}, \mathbf{R}_{t+1}(j,k), i = k$. The algorithm stops until the clustering centers remain unchanged after several iterations or the execution iteration number of the algorithm exceeds the predefined number of iterations.

Algorithm 2 The clustering algorithm.

```
Input: word embedding ${v(w)}_{n_1}$, bias parameter $p$,
       damping coefficient $\lambda$, iterations $T$.
Output: clustering center ${v(w)}_{n_2}$.
compute the similarity matrix $\mathbf{S}$, and initialize $\mathbf{R}$, $\mathbf{A}$.
repeat
    compute $r_t(i, k)$ and $a_t(i, k)$;
    $r_{t+1}(i, k)$=(1-$\lambda$)$r_{t+1}(i, k)$+$\lambda r_t(i, k)$;
    $a_{t+1}(i, k)$=(1-$\lambda$)$a_{t+1}(i, k)$+$\lambda a_t(i, k)$;
until $t$+1 > $T$.
```

Based on the above method, we can replace the text with the corresponding semantic words $\{sw_1, sw_2, \cdots\}$. Moreover, DSRM-DNN can update the semantic words dynamically when classifying the texts, and further improves the adaptive ability of the model. Details are described in Section 3.3.

3.2 Word feature construction

According to the method in Section 3.1, the selected words are saved in the DSRM-DNN. The model contains n semantic words $\{sw_1, sw_2, sw_3, \cdots, sw_n\}$, each of which has a unique index (that is, $sw_i \neq sw_j$, $i,j = 1,2,3,\cdots,n$ and $i\neq j$). To maintain the representation ability of textual vectors and accelerate the training of the classifier, we construct a fusion feature to replace semantic words in the DSRM-DNN. Therefore, each text in the datasets can be represented by a n-dimensional vector, the expression formula is:

$$x_i = M\_TF(sw_i) \tag{14}$$

where $M\_TF(\cdot)$ is the feature fusion function. Different from the traditional bag of words (BOW) using statistical features to represent all of the words, the statistical features are the statistics of the number, position and span of words in the text.

Word frequency is the number that a word appears in the text. The higher the word frequency is, the more important the word is. It is one of the commonly used word attributes for statistical features. For the calculation of word frequency factor fre_i, we adopt the nonlinear function as:

$$fre_i = \frac{n}{n+1} \tag{15}$$

where n is the number of appearance of the word sw_i. The nonlinear function has two advantages: one is that the word frequency factor is positively proportional to the word frequency; the other is that when the word frequency increases to a certain extent, the values of the word frequency factor will decrease, which conforms to the language reality.

Part-of-speech factor is a quantification of part-of-speech. Part-of-speech analysis of existing semantic words shows that most of them are nouns. Compared with nouns, verbs and adjectives have relatively little influence. Since different part-of-speech has different effect on text classification, we divide the words into three categories according to their parts-of-speech as follows:

$$pos_i = \begin{cases} 1, & w_i \in nouns. \\ 0.8, & w_i \in verbs.\ or\ adjectives. \\ 0.6, & w_i \in otherwise. \end{cases} \tag{16}$$

The processing function of word length factor is as following:

$$lenth_i = \frac{len_i}{max(len_i)} \tag{17}$$

where len_i is the word length of the i th word sw_i, and $max(len_i)$ is the maximum length of all words in the text.

The position of a word in the text is also of great value in judging its importance. Different words appear in different positions in the text, and their ability to express the theme of the text is often different. Words in the title can better reflect the theme of the text than words in the abstract, the first paragraph and the body. We define the function according to different positions:

$$loc_i = \begin{cases} 0.75, & w_i\ in\ title; \\ 0.67, & w_i\ in\ abstract\ or\ paragraph_{first}; \\ 0.5, & w_i\ in\ body\ part. \end{cases} \tag{18}$$

The word span factor can effectively reduce the influence of local words on full-text words. To improve the existing original calculation method, we define the word span factor formula as follows:

$$span_i = \frac{last_i - first_i + 1}{sum_i} \times \frac{d_i}{D_i} \tag{19}$$

where $first_i$ is the position the word sw_i first appears, $last_i$ is the position the word sw_i last appears, sum_i is the total number of words in the text, d_i is the number of paragraphs that contain the word sw_i, and D_i is the total number of paragraphs.

Here we construct the fusion feature to represent the semantic words, and the new feature is computed as follows:

$$\begin{aligned} M\_TF(sw_i) \quad = \quad & \alpha_1 fre_i + \alpha_2 pos_i + \alpha_3 lenth_i \\ & + \alpha_4 loc_i + \alpha_5 span_i \end{aligned} \tag{20}$$

where fre_i is the word frequency factor, pos_i is the part-of-speech factor, $lenth_i$ is the word length factor, loc_i is the word position factor, $span_i$ is the word span factor, α_1, α_2, α_3, α_4 and α_5 are the weights of feature factors. Therefore, according to (14) we can use $\{x_1, x_2, x_3, \cdots, x_n\}_j$ to represent T_j with n semantic words.

3.3 Dynamic representation

Since training set and test set are assigned randomly and semantic words in the DSRM-DNN are composed of the words in the training set, while the words extracted from the test set may not appear in the DSRM-DNN. If the words extracted from the test text do not appear in the DSRM-DNN (denoted as $word_{out-DSRM}$), we sparsely represent these words using other semantic words or non-semantic words ($sw_{in-DSRM}$ or $word_{in-DSRM}$). The objective function is as follows:

$$\underset{\alpha}{\text{argmin}} \, \| y - \alpha X \|_2^2 \quad s.t. \| \alpha \|_1 \leq \varepsilon \tag{21}$$

or

$$\underset{\alpha}{\text{argmin}} \, \| y - \alpha X \|_2^2 + \lambda \| \alpha \|_1 \tag{22}$$

where y is the sample that needs to be reconstructed, X is a matrix of the embedding vectors, ε and λ are both small positive constants.

Although l_1-norm plays an implicit role in the selection of training samples in regression, the computational cost of the iterative solution is very high, so we replace the regularization term with l_2-norm. The objective function can be expressed as:

$$\underset{\alpha}{\text{argmin}} \, \| y - \alpha X \|_2^2 + \lambda \| \alpha \|_2^2 \tag{23}$$

To improve the efficiency of sparse representation, new semantic representation words can be divided into the following two situations:

$$\begin{cases} \text{argmin}_{x_i} \| k_i - x_i K \|_2^2 + \lambda \| x_i \|_2^2, \ m_1/(m_1 + m_2) < \alpha, \\ \text{argmin}_{y_i} \| k_i - y_i W \|_2^2 + \lambda \| y_i \|_2^2, m_1/(m_1 + m_2) \geq \alpha \end{cases} \tag{24}$$

where λ is the weight parameter, k_i is the i th semantic word in the $word_{out-DSRM}$, $x_i \in \mathbb{R}^{m_2}$ is the reconstruction vector, $K \in \mathbb{R}^{m_2 \times n}$ is composed of m_2 semantic words in $sw_{in-DSRM}$, $y_i \in \mathbb{R}^{m_3}$ is the reconstruction vector, $W \in \mathbb{R}^{m_3 \times n}$ is composed of m_3 non-semantic words in $word_{in-DSRM}$, and $m_1 = num(word_{out-DSRM})$, $m_2 = num(sw_{in-DSRM})$, $m_3 = num(word_{in-DSRM})$. Besides, new semantic words in the test texts (denoted as $sw_{out-DSRM}$) can be added to DSRM-DNN to further improve the representation and adaptability of DSRM-DNN.

3.4 Classifier construction

BP network is multilayer feedforward neural network based on error backward propagation algorithm. The principle is to calculate the difference between the actual output and the expected output recursively, and then adjust the weights according to the difference. In order to reduce the computational cost of neural network training, we construct a BP network including an input layer, an output layer and two hidden layers, as shown in Fig. 3. We input the obtained feature matrix $H_{m \times n}$ to the network for training. The activation function and its derivative are given as follows:

$$S(x) = \frac{1}{1 + e^{-x}} \tag{25}$$

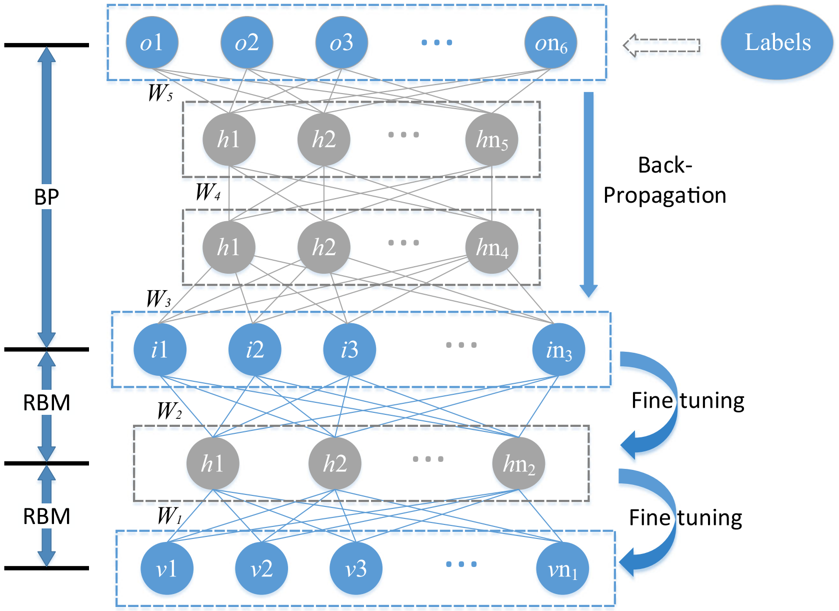
$$S'(x) = \frac{e^{-x}}{(1 + e^{-x})^2} = S(x)(1 - S(x)) \tag{26}$$

Since the randomly initialized parameters may make the network convergence to the local optimum and reduce the training effect, this paper adopts the deep belief network (DBN) co-constructed by BP network and restricted Boltzmann machine [2] to initialize the parameters of BP network.

Algorithm 3 Restricted Boltzmann machine

```
Input: data $H_{m \times n}$, training cycle $K$, learning rate $\varepsilon$,
       hidden layer elements $u$.
Output: weight matrices $W$, bias vector $a$,$b$.
randomly initialize $W$,$a$,$b$.
for $o$ = 1 to $K$ do
    for $t$ = 1 to $m$ do
        initialize $v^{(0)}$,$v^{(1)}$,$h^{(0)}$;
        for $i$ = 1 to $u$ and $j$ = 1 to $n$ do
            $\Delta w_{i,j} = \Delta w_{i,j} + [P(h_i = 1|v^{(0)})v_j^{(0)}$ - $P(h_i = 1|v^{(1)})v_j^{(1)}]$;
            $\Delta a_j = \Delta a_j + [v_j^{(0)}$ - $v_j^{(1)}]$;
            $\Delta b_i = \Delta b_i + [P(h_i = 1|v^{(0)})$ - $P(h_i = 1|v^{(1)})]$.
        end
    end
    $W = W + \varepsilon(\frac{1}{m}\Delta W)$,$a = a + \varepsilon(\frac{1}{m}\Delta a)$,$b = b + \varepsilon(\frac{1}{m}\Delta b)$.
end
```

Fig. 3 The structure diagram of BP network and DBN



Each restricted Boltzmann machine has n visible elements and m hidden elements, and the states of visible elements and hidden elements are represented by vectors a and b respectively, then the energy function is:

$$\mathrm{E}(v,h,\theta)=-\sum_{i=1}^{n}\sum_{j=1}^{m}v_iW_{ij}h_j-\sum_{i=1}^{n}a_iv_i-\sum_{j=1}^{m}b_jh_j \tag{27}$$

where $\theta=\{W_{ij},a_i,b_j\}$ is the parameter of RBM, W_{ij} is the connection weight between visible element i and hidden element j, a_i and b_j are the bias of element i and j. We get the joint probability distribution function and likelihood function of v and h:

$$\mathrm{p}(v,h|\theta)=\mathrm{e}^{-\mathrm{E}(v,h|\theta)}/\mathrm{Z}(\theta),\quad \mathrm{Z}(\theta)=\sum_{v,h}\mathrm{e}^{-\mathrm{E}(v,h|\theta)} \tag{28}$$

$$\mathrm{p}(v|\theta)=\sum_{h}\mathrm{e}^{-\mathrm{E}(v,h|\theta)}/\mathrm{Z}(\theta) \tag{29}$$

where $\mathrm{Z}(\theta)=\sum_{v,h}\mathrm{e}^{-\mathrm{E}(v,h|\theta)}$ is the normalization factor.

4 Experiments

In this section, we introduce the experiments in detail and report the results of DSRM-DNN and comparative methods on benchmark datasets. To comprehensively analyze the performance of the proposed methods, we mainly perform five experiments. The first experiment studies the related setting of semantic words extraction, including the dimension of word embedding, the size of the context window, and the number of semantic words. In the second and third experiments, we analyze the effects of different word attributes and dynamic representation. The fourth experiment compares the performance of different classification methods. Finally, we test DSRM-DNN and compare it with nine baselines respectively on four datasets Fig. 3.

4.1 Datasets

We use four representative datasets to evaluate the proposed method as shown in Table 2. The datasets are split into training sets and test sets, and 80% of the randomly selected texts are used to train classifiers and the remaining texts are used to verify the effect of text classification methods.

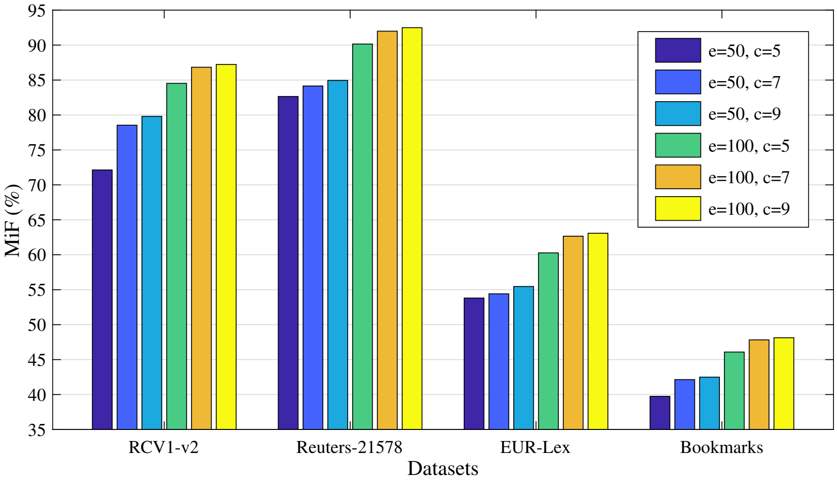
4.2 Baselines

To make the experimental comparison more comprehensive and objective, we not only use the traditional machine

Table 2 Datasets statistics. D_num is the number of total texts, L_num is the number of labels, Vo_size is the size of total vocabulary, L_ave is the average number of labels per text, Tr_num is the number of train texts, Te_num is the number of test texts

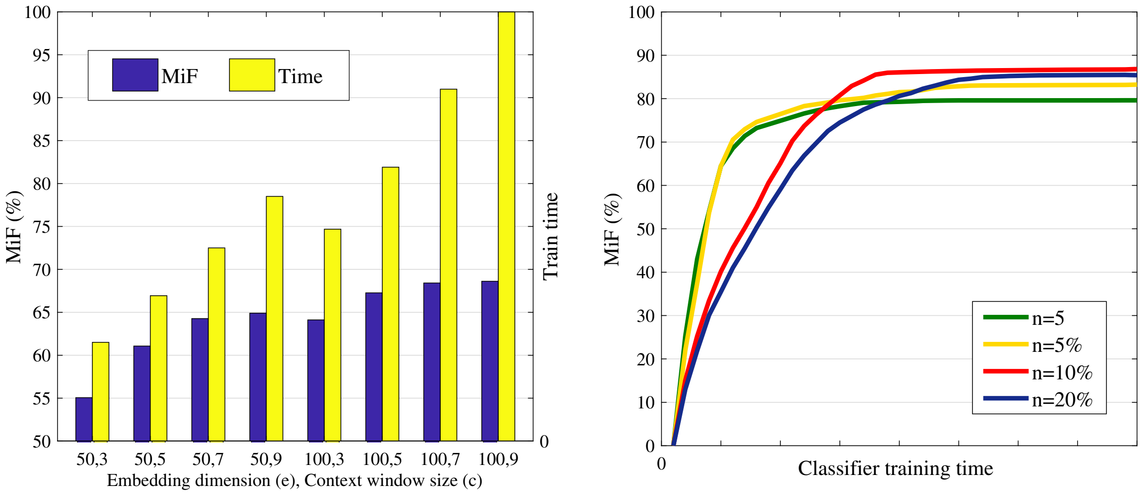
Datasets	D_num	L_num	Vo_size	L_ave	Tr_num	Te_num
RCV1-v2	804414	103	47236	3.24	723531	160883
EUR-Lex	19348	3993	26575	5.32	15478	3870
Reuters-21578	10789	90	18637	1.13	8631	2158
Bookmarks	87856	208	2150	2.03	70285	17571

Fig. 4 The influence of word embedding and context window, e is the dimension of word embedding, c is the size of context window



learning algorithms such as decision tree, k-nearest neighbor, but also use the latest methods such as neural networks for comparative analysis. The details of these methods are as follows:

- **Multi-label decision tree (ML-DT)** [31] adopts decision tree techniques to process multi-label data, where an information gain criterion based on multi-label entropy is utilized to build the decision tree recursively.
- **Multi-label k-nearest neighbor (ML-KNN)** [30] adopts k-nearest neighbor techniques to process multi-label data, where maximum posterior probability is utilized to predict the labels of nearest samples.
- **Binary relevance (BR)** transforms the multi-label learning problem into multiple independent binary classification problems, where each binary classification problem corresponds to a possible label in the label space.
- **Classifier chains (CC)** transforms the multi-label learning problem into a chain of binary classification problems, in which subsequent binary classifiers in the chain build on the predictions of preceding ones. In BR and CC, we use the Euclidean-SVMs [24] as the base classifiers.
- **Multi-label neural networks (ML-NN)** [28] represents the multi-label classification problem as a neural network with multiple output nodes. Each label in the dataset corresponds to an output node of the network, and the output layer can model the dependencies between different categories.
- **Hierarchical ARAM neural network (HARAM)** [27] is an extension of the fuzzy Adaptive Resonance Associative Map (ARAM), to speed up the classification on high-dimensional and large-scale datasets.
- **Convolutional and recurrent neural networks (CNN-RNN)** [29] utilizes the ensemble application of convolutional and recurrent neural networks to capture the



(a) (b)

Fig. 5 The train time of word embedding and classifier on RCV1-v2

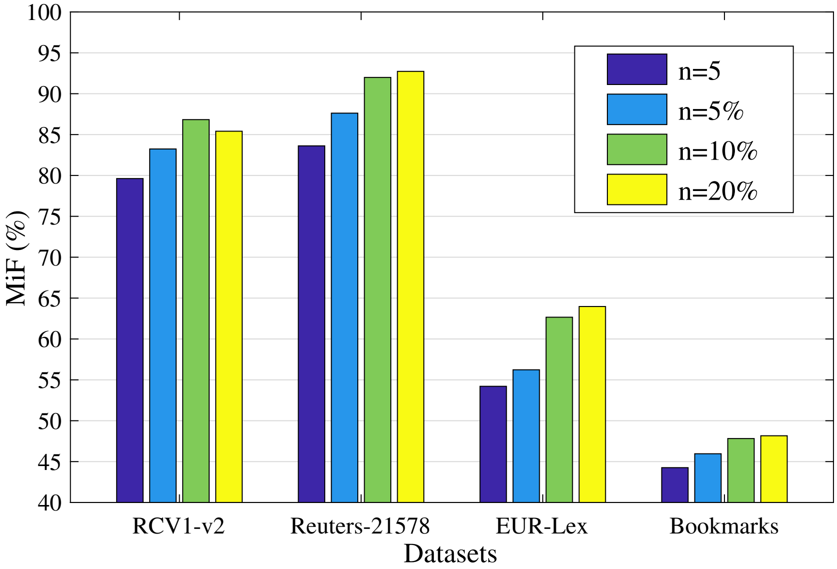


Fig. 6 The influence of semantic words number, and n is the number of semantic words

global and local textual semantics, and to model label correlations while having a tractable computational complexity.

- **Hierarchical label set expansion (HLSE)** [19] regularizes data labels and considers extreme multi-class and multi-label text classification when defining hierarchical label set.
- **Supervised representation learning (SERL)** [25] is the framework based on neural networks, which can learn global feature representation by jointly considering all labels in an effective supervised manner.

4.3 Evaluation metrics

For evaluating the performance of multi-label classifiers, we use One-error, Hamming Loss, Rank Loss and Micro/Macro-averaged Precision, Recall, F1-Score to test the efficiency of the experiment. The details are as follows:

- **One-error (O-error)** shows the proportion that the label with the highest ranking score is not in the correct label set.
- **Hamming loss (H-Loss)** computes the symmetric difference between the predicted labels and the relevant labels and calculates the fraction of its difference in the label space.
- **Ranking loss (R-Loss)** evaluates the fraction of reversely ordered label pairs, e.g. an irrelevant label is ranked higher than a relevant label.
- **Precision, Recall and F1-score** are calculated based on the number of true positives (*tp*), true negatives (*tn*), false positives (*fp*) and false negatives (*fn*). There are two ways to calculate these metrics over the whole test data: Micro-

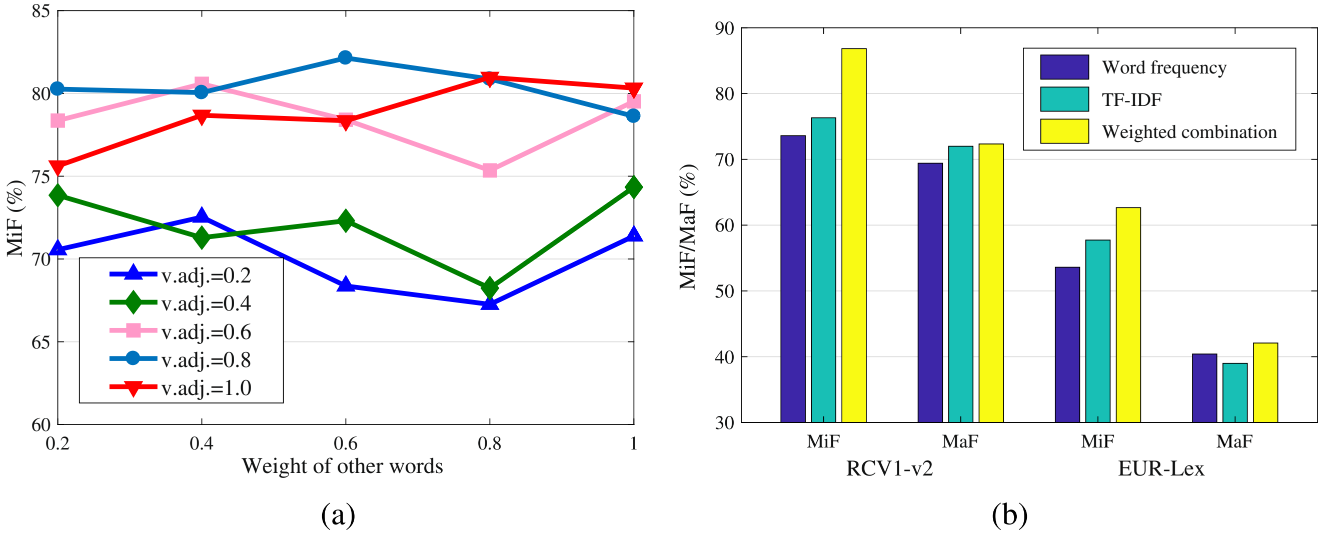


Fig. 7 The study of word feature construction. **a** We set the weight of the noun to 1 and seek the optimal combination by adjusting the weights of the verb, adjective and other words. **b** The performance of different word attributes on the final classification result

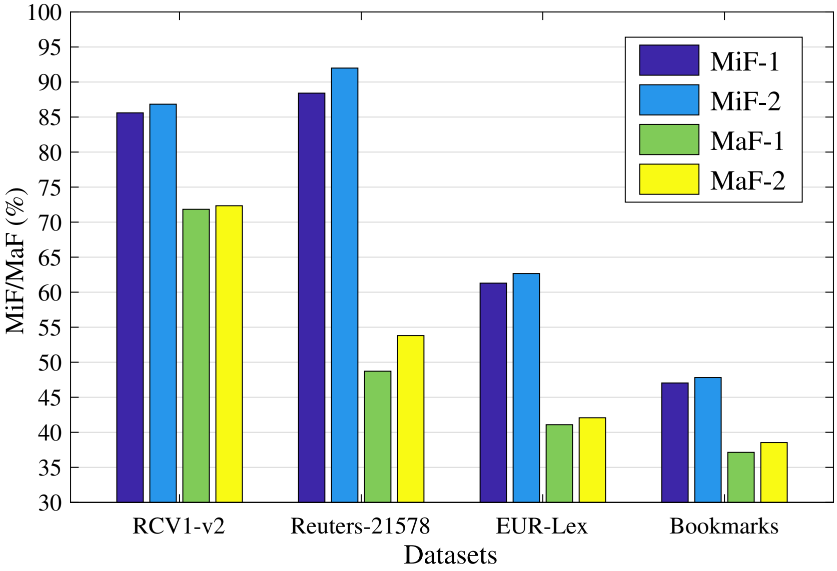


Fig. 8 The influence of dynamic representation, MiF-1/MaF-1 is the result without dynamic representation, MiF-2/MaF-2 is the result of the proposed method

averaged and Macro-averaged. The former counts all true positives, true negatives, false positives and false negatives first among all labels and then has a binary evaluation for its overall counts, while the latter refers to the average performance (Precision, Recall and F1-score) over labels. To be specific, the computations of Micro/Macro-averaged Precision, Recall, F1-score are illustrated below:

$$\begin{aligned} P_{micro} &= \frac{\Sigma_{l=1}^{L} tp_l}{\Sigma_{l=1}^{L} tp_l + fp_l}, \quad R_{micro} = \frac{\Sigma_{l=1}^{L} tp_l}{\Sigma_{l=1}^{L} tp_l + fn_l}, \\ F1_{micro} &= \frac{\Sigma_{l=1}^{L} 2tp_l}{\Sigma_{l=1}^{L} 2tp_l + fp_l + fn_l} \\ P_{macro} &= \frac{1}{L}\Sigma_{l=1}^{L} \frac{tp_l}{tp_l + fp_l}, \quad R_{macro} = \frac{1}{L}\Sigma_{l=1}^{L} \frac{tp_l}{tp_l + fn_l}, \\ F1_{macro} &= \frac{1}{L}\Sigma_{l=1}^{L} \frac{2tp_l}{2tp_l + fp_l + fn_l} \end{aligned}$$

4.4 Experimental results and analysis

4.4.1 Semantic words extraction

In the process of semantic words extraction, we analyze the effect of embedding dimension and context window size by fixing other variables. Note that we set the minimum word frequency of the input embedding model to 3 (that is, words that occur more than three times in the text are trained to generate the embedding vectors). The setting reduces the influence of uncommon words on the training process and improves the training speed of the word embedding model.

The results show that the Micro-averaged F1-Score (MiF) with word embedding dimension e = 100 is significantly higher than the MiF with e = 50, which indicates that the beneficial semantic information of embedding vectors increases as embedding dimension does. As shown in Fig. 4, the size of the context window has less impact on the classification performance than the dimension of word embedding. Besides, the increment of MiF decreases gradually with the increase of the context window, which indicates that the correlation of words decreases rapidly with the increase of distance between them. Note that in Fig. 5a the increase of vector dimension or context window enriches the semantic information of the word embedding, but the training time also increases greatly.

To obtain a reasonable number of semantic words, we set the number of semantic words in each text to 5, or 5%, 10%, 20% of the number of words in the text. In general, with the increase of selected words from each text, the final results are gradually improved, because the semantic representation ability of text vectors is more suitable for the training of classifiers. On RCV1-v2, the MiF with n = 20% is significantly lower than that with n = 10%. The reason is that DSRM-DNN contains most of the words that are conductive to classification when n = 10%. As n continues to increase, the newly selected words are irrelevant and interfere with the performance of the existing words, so the increase of semantic words reduces the training effect of the classifier. The experimental results in Figs. 5b and 6 show that the proposed method obtains excellent accuracy when n = 10% and 20% respectively, but the training time of classifier increases rapidly with the growth of DSRM-DNN.

4.4.2 Word feature construction

There are many ways to quantify words in NLP, among which TF-IDF and word frequency are widely used. In the experiment, we test the influence of word attributes on RCV1-v2, and make a quantitative analysis of the representation ability of part-of-speech.

As shown in Fig. 7a, we test several possible combinations of part-of-speech weights. Note that part-of-

Table 3 Comparison of different methods on benchmark datasets

		Evaluation Metrics							
Dataset	Method	H/R-loss	O-error	MiP	MiR	MiF	MaP	MaR	MaF
RCV1-v2	ML-DT	0.0102	0.0393	0.6347	0.7497	0.6874	0.3961	0.5626	0.4649
	ML-KNN	0.0081	0.0287	0.7712	0.7836	0.7774	0.5741	0.5676	0.5708
	BR	0.0088	0.0125	0.8984	*0.8125*	*0.8533*	0.7946	0.6237	0.6989
	CC	0.0093	0.0454	0.8863	0.8114	0.8472	0.7682	0.6449	0.7012
	HARAM	0.0074	0.0215	0.9046	0.7824	0.8391	0.7756	0.5693	0.6566
	BP-MLL	0.0089	0.0349	0.6685	0.7695	0.7155	0.4385	0.5803	0.4995
	CNN-RNN	0.0086	**0.0102**	0.8895	0.8150	0.8506	*0.8034*	**0.6465**	*0.7165*
	HLSE	0.0079	0.0132	0.8673	0.7736	0.8178	0.7825	0.5876	0.6712
	SERL	0.0081	0.0245	0.9137	0.7984	0.8522	0.8015	0.6215	0.7001
	DSRM-DNN-1	*0.0076*	*0.0119*	*0.9124*	0.8062	0.8560	0.8164	*0.6413*	0.7183
	DSRM-DNN-2	**0.0072**	0.0109	**0.9196**	**0.8225**	**0.8683**	**0.8326**	0.6395	**0.7234**
Reuters-21578	ML-DT	0.0054	0.1698	0.7876	0.8096	0.7984	0.3510	0.2806	0.3119
	ML-KNN	0.0088	0.4742	0.8038	0.4734	0.5959	0.3422	0.2056	0.2569
	BR	0.0032	0.0911	0.9403	0.8239	0.8782	0.4645	0.3507	0.3997
	CC	*0.0031*	0.0844	0.9374	0.8286	0.8796	0.4706	0.3613	0.4088
	HARAM	0.0066	0.1127	0.7480	0.7756	0.7616	0.2981	0.2424	0.2674
	BP-MLL	0.0049	0.0868	0.7876	0.8616	0.8229	*0.4809*	*0.4761*	*0.4785*
	CNN-RNN	0.0038	**0.0836**	0.9025	0.8131	0.8555	0.3697	0.2875	0.3235
	HLSE	0.0047	0.0984	0.9258	0.8215	0.8705	0.4582	0.3974	0.4256
	SERL	0.0053	0.1332	0.9431	*0.8684*	*0.9042*	0.4796	0.4520	0.4654
	DSRM-DNN-1	0.0029	0.0842	*0.9426*	0.8791	0.9097	0.4913	**0.4831**	0.4872
	DSRM-DNN-2	**0.0025**	*0.0855*	**0.9585**	**0.8842**	**0.9199**	**0.6147**	0.4785	**0.5381**
EUR-Lex	ML-DT	0.0894	0.6060	0.2113	0.5495	0.3052	0.1567	0.2254	0.1849
	ML-KNN	0.0572	0.3052	0.3913	0.1716	0.2386	**0.5141**	0.2318	0.3195
	BR	0.0642	0.1918	0.6124	0.4945	0.5472	0.4260	0.3643	0.3927
	CC	0.0386	0.7329	0.1508	0.5426	0.2360	0.2618	0.3012	0.2801
	HARAM	*0.0164*	*0.1851*	**0.6735**	0.4777	0.5589	0.4158	0.3271	0.3662
	BP-MLL	0.0209	0.2258	0.3294	0.4138	0.3668	0.2331	0.3063	0.2647
	CNN-RNN	0.0193	0.2014	0.6226	0.4710	0.5363	0.3727	0.3103	0.3387
	HLSE	0.0284	0.2134	0.6387	0.5576	0.5954	0.3854	0.3942	0.3898
	SERL	0.0182	0.2349	*0.6668*	*0.5761*	0.6181	0.4085	0.3853	*0.3966*
	DSRM-DNN-1	0.0153	**0.1632**	0.6423	0.5861	*0.6129*	*0.4298*	*0.3932*	0.4109
	DSRM-DNN-2	**0.0134**	0.1714	0.6714	**0.5874**	**0.6266**	0.4315	**0.4107**	**0.4208**
Book marks	ML-DT	0.1084	0.5598	0.1153	0.2067	0.1480	0.1324	0.1458	0.1388
	ML-KNN	0.0964	0.5574	0.3143	0.3518	0.3320	0.2514	0.2780	0.2640
	BR	0.0993	0.5318	0.2821	0.2546	0.2676	0.1950	0.1880	0.1914
	CC	0.0895	0.5428	0.1922	0.3967	0.2589	0.1642	0.3104	0.2148
	HARAM	0.0707	*0.4876*	0.3924	0.3951	0.3937	0.3614	0.3409	0.3509
	BP-MLL	0.0684	0.5589	0.0943	*0.5682*	0.1618	0.1115	0.2743	0.1586
	CNN-RNN	0.0629	**0.4732**	*0.3929*	0.4096	0.4011	0.2257	0.3321	0.2688
	HLSE	0.0782	0.5349	0.3658	0.4627	0.4086	0.3472	0.3274	0.3370
	SERL	0.0879	0.4974	0.3817	0.5232	*0.4414*	*0.3715*	0.3603	*0.3658*
	DSRM-DNN-1	*0.0634*	0.4926	0.3974	**0.5762**	0.4704	0.3841	*0.3596*	0.3714
	DSRM-DNN-2	**0.0572**	0.4828	**0.4123**	0.5691	**0.4782**	**0.4018**	**0.3702**	**0.3854**

DSRM-DNN-1/2 are the proposed methods (without the dynamic representation/with the dynamic representation). To observe the experimental results more conveniently, we bold the optimal results, underline the sub-optimal ones, and mark the third optimal results in italics speech cannot quantize the text alone (quantifying the word using only part-of-speech results in very poor performance), we analyze its effect indirectly through its weighted combination with word frequency. The experimental results show that (1, 0.8, 0.6) is the optimal weight combination of parts-of-speech. Besides, the results in Fig. 7b show that the weighted combination is more effective than single word attribute.

4.4.3 Dynamic representation

To verify the effectiveness of the dynamic representation method, we test MiF and Macro-averaged F1-Score (MaF) of different classification methods on four datasets.

As shown in Fig. 8, the dynamic representation method improves the performance of the classifier on most datasets. The performance of the optimization on RCV1-v2 is poor, the reason is that existing DSRM-DNN contains most of the representative words, and the new words added by the dynamic representation is less. The high average number of labels per text on EUR-Lex results in the selected semantic words being very important and increases the impact of dynamic representation, so the proposed method can achieve better performance.

4.4.4 Comparison of different methods

We test different classification methods on the datasets mentioned in Section 4.1 and use the various evaluations to analysis their performance. As the performance of different methods using Hamming Loss to evaluate on the above datasets is not obvious, we replace Hamming Loss with Rank Loss on EUR-Lex and Bookmarks. In the experiments, we set the parameters of DSRM-DNN as e = 100, c = 7, n = 10%.

Both BR and CC in Table 3 are based on Euclidean-SVMs, and achieve significantly higher recall and precision than ML-DT and ML-KNN. It shows that the problem transformation methods are more effective than the algorithm adaptation methods for these datasets. Among the methods based on deep learning, DSRM-DNN has better classification effects on RCV1-v2, which shows that the proposed method is more suitable for multi-label text classification. In addition, we observe that many classification methods perform well on Reuter-21578. The reason is that the weak correlation of labels leads to the great difference between the texts from different categories. The label space of EUR-Lex and Bookmarks in Table 2 is complex and difficult to be classified. The results show that the proposed method also outperforms the existing methods.

The experimental results also show that the performance of the classifiers based on deep learning is obviously better than that of the classifiers traditional machine learning algorithms. DSRM-DNN is optimal or sub-optimal on most datasets, and its classification results are better than the compared classification methods. Besides, our method can effectively reduce the negative impact of lowly-frequent words and new words, so it is more suitable for datasets with strong label correlation and wide semantic word distribution.

5 Conclusion

In this paper, we propose a DSRM-DNN method to extract semantic words and construct text features, and integrate multiple restricted Boltzmann machines to construct deep belief networks to initialize the classifier and accelerate its convergence. Comparison with the state-of-the-art methods on four datasets demonstrates the proposed method achieves better results on both text quantization and multi-label classification. To further improve the speed and accuracy of text classification, we will focus on the optimization of semantic extraction and the neural network construction in the future research.

Acknowledgements This work is supported by the National Natural Science Foundation of China (No. 61702310), the major fundamental research project of Shandong, China (No.ZR2019ZD03), and the Taishan Scholar Project of Shandong, China.

References

1. Hassan A, Mahmood A (2017) Efficient deep learning model for text classification based on recurrent and convolutional layers[C]. In: International conference on machine learning and applications (ICMLA), pp 1108–1113
2. Pacheco A G C, Krohling R A, da Silva C A S (2018) Restricted Boltzmann machine to determine the input weights for extreme learning machines[J]. Expert Syst Appl 96:77–85
3. Shin K, Abraham A, Han SY (2006) Improving KNN text categorization by removing outliers from training set[C]. In: International conference on intelligent text processing and computational linguistics, pp 563–566
4. Ali S A, Sulaiman N, Mustapha A, et al. (2009) Decision tree response classification[J]. Inf Technol J 8(8):1256–1262
5. Rinaldi AM (2008) A content-based approach for document representation and retrieval[C]. In: Proceedings of the 8th ACM symposium on document engineering, pp 106–109
6. Shang C, Li M, Feng S et al (2013) Feature selection via maximizing global information gain for text classification[J]. Knowl-Based Syst 54:298–309
7. Bai X, Shi B, Zhang C, et al. (2017) Text/non-text image classification in the wild with convolutional neural networks[J]. Pattern Recogn 66:437–446
8. Zhu X, Vondrick C, Fowlkes C C, et al. (2016) Do we need more training data?[J]. Int J Comput Vis 119(1):76–92
9. Bui D D A, Fiol G D, Jonnalagadda S (2016) PDF text classification to leverage information extraction from publication reports[J]. J Biomed Inform 61:141–148
10. Shang F, Zhang H, Sun J, et al. (2019) Semantic consistency cross-modal dictionary learning with rank constraint[J]. J Vis Commun Image Represent 62:259–266

11. Al-Salemi B, Noah S A M, Aziz M J A (2016) RFBoost: An improved multi-label boosting algorithm and its application to text categorisation[J]. Knowl-Based Syst 103:104–117
12. Wang P, Xu B, Xu J, et al. (2016) Semantic expansion using word embedding clustering and convolutional neural network for improving short text classification[J]. Neurocomputing 174:806–814
13. Abualigah L M, Khader A T (2017) Unsupervised text feature selection technique based on hybrid particle swarm optimization algorithm with genetic operators for the text clustering[J]. J Supercomput 73(11):4773–4795
14. Wu L, Hoi S C H, Yu N (2010) Semantics-preserving bag-of-words models and applications[J]. IEEE Trans Image Process 19(7):1908–1920
15. Singh D, Singh B (2019) Hybridization of feature selection and feature weighting for high dimensional data[J]. Appl Intell 49(4): 1580–1596
16. Abualigah L M, Khader A T, Hanandeh E S (2018) Hybrid clustering analysis using improved krill herd algorithm[J]. Appl Intell 48(11):4047–4071
17. Abualigah L M, Khader A T, Hanandeh E S (2018) A combination of objective functions and hybrid Krill herd algorithm for text document clustering analysis[J]. Eng Appl Artif Intel 73:111–125
18. Abualigah L M, Khader A T, Hanandeh E S et al (2017) A novel hybridization strategy for krill herd algorithm applied to clustering techniques[J]. Appl Soft Comput 60:423–435
19. Gargiulo F, Silvestri S, Ciampi M et al (2019) Deep neural network for hierarchical extreme multi-label text classification[J]. Appl Soft Comput 79:125–138
20. Yu B, Xu Z (2008) A comparative study for content-based dynamic spam classification using four machine learning algorithms[J]. Knowl-Based Syst 21(4):355–362
21. Liu H, Xu B, Lu D, et al. (2018) A path planning approach for crowd evacuation in buildings based on improved artificial bee colony algorithm[J]. Appl Soft Comput 68:360–376
22. Liu H, Liu B, Zhang H, et al. (2018) Crowd evacuation simulation approach based on navigation knowledge and two-layer control mechanism[J]. Inform Sci 436:247–267
23. Shang F, Zhang H, Zhu L, et al. (2019) Adversarial cross-modal retrieval based on dictionary learning[J]. Neurocomputing 355:93–104
24. Lee L H, Wan C H, Rajkumar R, et al. (2012) An enhanced support vector machine classification framework by using euclidean distance function for text document categorization[J]. Appl Intell 37(1):80–99
25. Huang M, Zhuang F, Zhang X, et al. (2019) Supervised representation learning for multi-label classification[J]. Mach Learn 108(5): 747–763
26. de Campos Ibáñez LM, Romero AE (2009) Bayesian network models for hierarchical text classification from a thesaurus[J]. Int J Approx Reason 50(7):932–944
27. Benites F, Sapozhnikova E (2015) Haram: A hierarchical aram neural network for large-scale text classification[C]. In: 2015 IEEE international conference on data mining workshop (ICDMW), pp 847–854
28. Zhang M L, Zhou Z H (2006) Multilabel neural networks with applications to functional genomics and text categorization[J]. IEEE Trans Knowl Data Eng 18(10):1338–1351
29. Chen G, Ye D, Xing Z et al (2017) Ensemble application of convolutional and recurrent neural networks for multi-label text categorization[C]. In: 2017 international joint conference on neural networks (IJCNN), pp 2377–2383
30. Zhang M L, Zhou Z H (2007) ML-KNN: A lazy learning approach to multi-label learning[J]. Pattern Recognit 40(7):2038–2048
31. Clare A, King RD (2001) Knowledge discovery in multi-label phenotype data[C]. In: European conference on principles of data mining and knowledge discovery, pp 42–53
32. Liu L, Zhang B, Zhang H et al (2019) Graph steered discriminative projections based on collaborative representation for image recognition[J]. Multimed Tools Appl 78(17):24501–24518
33. Zhou S, Li K, Liu Y (2009) Text categorization based on topic model[J]. Int J Computat Intell Syst 2 (4):398–409

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.