Text Classification Based on Word2vec and Convolutional Neural Network

Lin Li, Linlong Xiao, Wenzhen Jin, Hong Zhu, and Guocai Yang(✉)

School of Computer and Information Science, Southwest University, Chongqing, China
cqkxxn@163.com, paul.g.yang@gmail.com

Abstract. Text representations in text classification usually have high dimensionality and are lack of semantics, resulting in poor classification effect. In this paper, TF-IDF is optimized by using optimization factors, then word2vec with semantic information is weighted, and the single-text representation model CD_STR is obtained. Based on the CD_STR model, the latent semantic index (LSI) and the TF-IDF weighted vector space model (T_VSM) are merged to obtain a fusion model, CD_MTR, which is more efficient. The text classification method MTR_MCNN of the fusion model CD_MTR combined with convolutional neural network is further proposed. This method first designs convolution kernels of different sizes and numbers, allowing them to extract text features from different aspects. Then the text vectors trained by the CD_MTR model are used as the input to the improved convolutional neural network. Tests on two datasets have verified that the performance of the two models, CD_STR and CD_MTR, is superior to other comparable textual representation models. The classification effect of MTR_MCNN method is better than that of other comparison methods, and the classification accuracy is higher than that of CD_MTR model.

Keywords: Text classification · Text representation · Word2vec
Convolutional neural network

1 Introduction

Most of the data generated by Internet are stored in a format of text, and text data occupy an important position. Manually organizing and managing textual information has been unable to adapt to the ever-expanding digital information of the Internet age. With such a large amount of data and a variety of data forms, finding the right method to effectively manage and use these text data is very important. Efficient feature extraction and text representation are challenges for text classification, and it is also the first problem that should be solved in text classification.

Most of the early text representation methods used were vector space models. Later, most researchers used the distributed representation of words [1], which was proposed by Hinton in 1986 and could overcome the shortcomings of the one-hot representation. Bengio proposed to use a three-layer neural network to train text representation model in 2003 [2]. Hinton used hierarchical ideas in 2008 to improve the training process from

L. Cheng et al. (Eds.): ICONIP 2018, LNCS 11305, pp. 450–460, 2018.
https://doi.org/10.1007/978-3-030-04221-9_40

the hidden layer to the output layer in the Bengio method, speeding up the training model [3]. Mikolov proposed a neural network model to train distributed word vectors in 2013. The training tool word2vec implemented by this model has been widely used [4, 5]. Hu used a convolutional neural network to extract semantic combination information from local words in a sentence through a neural network in 2014 [6].

We find traditional text representation methods, such as Boolean models and vector space models, have problems of data sparseness and dimensional disaster. With the rapid development of machine learning and deep learning technologies, researchers have begun to use various neural network to construct text representation models and map texts to low-dimensional continuous vectors through neural network, improving the model's representation ability. However, the existing neural network text representation model also has some problems. First of all, though the neural network obtains better semantic information for the text representation, its class distinction ability is lacking. Secondly, the existing method for extracting text features using convolutional neural network is based on the length of the longest text in the data set. Texts that are shorter than this length are filled with special characters. The introduction of too many non-semantic characters in this text affects the original information of the text and results in poor classification effect.

2 Related Work

Text representation is an important step in text classification. The original texts are unstructured data. You must find a suitable representation method to convert the text content into information that computer can recognize. The text representation mainly contains two aspects: representation and calculation, respectively referring to definition of feature selection and feature extraction, and the definition of computational weighting and semantic similarity [7]. The Vector Space Model (VSM) was proposed by Salton in the 1970 [8]. It simplifies the process of processing text content into vector operations in vector space, and expresses the similarity of text semantics by calculating spatial similarity. VSM is a common and very classic text representation. But the problem of dimension disaster exists in vector space model.

The Convolutional Neural Network (CNN) is a deep neural network that has made major breakthroughs in computer vision and speech recognition. It is widely used in image understanding [9, 10]. In recent years, continuous development of convolutional neural network has been used in natural language processing tasks such as text classification and element identification [11]. Wang proposed a semi-supervised convolutional neural network to enhance the semantic relevance of the context [12]. The c-lstm model proposed by Zhou, c-lstm uses the convolutional neural network to extract text sentence features and uses short-term memory recursive neural network to obtain sentence representations [13]. Lai proposed recursive convolutional neural network for text classification, which introduces less noise than traditional neural network [14].

3 Methods

3.1 Word2vec

Word2vec can train word vectors quickly and efficiently. There are two Word2vec models, CBOW model and Skip-gram model. The CBOW model uses the c words before and after the word w(t) to predict the current word; whereas the Skip-gram model does the opposite. It uses the word w(t) to predict the c words before and after it. The two model training methods are respectively shown in Fig. 1 left and right. This paper uses the CBOW model to train word vectors.

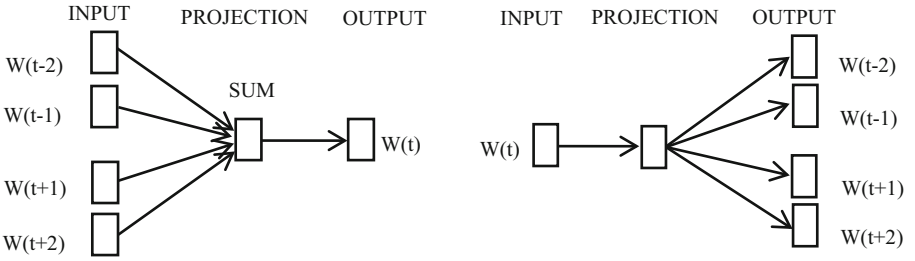

Fig. 1. CBOW and Skip-gram

3.2 Convolutional Neural Network

Convolution Layer

The convolutional layer is also called feature extraction layer. This layer is the core part of the convolutional neural network and can describe the local characteristics of the input data. The convolution kernel $w \in Q^{hk}$ included in the convolution operation [11] will generate a new feature value each time it passes through a word sequence window with a height of h and a width of k. For example, a feature point c_i in a feature map is the result of the window $x_{i:i+h-1}$ after convolution operation, that is, each feature value can be obtained by formula 1:

$$c_i = f(w \cdot x_{i:i+h-1} + b) \tag{1}$$

Where, $x_i \in Q^k$, w is the weight parameter of the convolution kernel; b is the offset term of the convolution layer; and f is a nonlinear activation function.

When training a convolutional neural network, it is necessary to establish a convolution kernel sliding stride, which can be set to be 1, 2 or more. The convolution kernel can convolve the input data to get a feature map, as shown in 2:

$$\mathrm{c} = [c_1, c_2, \cdots, c_{s-h+1}] \tag{2}$$

Pooling Layer

The pooling layer is generally disposed between two consecutive convolution layers. The pooling layer down-samples the feature map of the convolutional layer output, aggregates the statistics of all the feature maps of the convolutional layer, simplifies the information output from the convolutional layer through the pooling layer, and reduces the features and network parameters.

3.3 The Idea of the CD_STR Model

The main idea of IDF in TF-IDF algorithm is: if there are fewer documents containing characteristic words t, larger IDF indicates that characteristic words t have better category discrimination ability. The simple structure of IDF in TF-IDF cannot effectively reflect the importance of words and the distribution of feature words. The CD_STR model first considers that if a word appears in each text, and the frequency of occurrence in each text or in each type of text does not differ much, then the word contributes very little to the category distinction and should be filtered out or given a smaller weight. Conversely, the feature words should be given a higher weight value.

If there are three characteristic words t_1, t_2, t_3, in the three categories c_1, c_2, c_3, the distributions are (8, 8, 8), (5, 8, 5), (1, 8, 5). Then, the weights of these three feature words should be increased successively, because the frequency of t_3 in each category is relatively uneven compared to t_1, t_2. So, it will be better to distinguish categories.

CD_STR optimizes the TF-IDF algorithm mainly based on the distribution of feature words in various category, and specific improved algorithm is shown in formula 3:

$$G(t,\, d) \;=\; \frac{tf_{ij}(t,d) \;\times\; idf_i(t) \;\times\; \sum_k p(t|\, c_k)^2\, p(c_k\,|t)^2}{\sqrt{\sum_{i=1}^{n} [tf_{ij}(t,d) \;\times\; idf_i(t)]^2}} \tag{3}$$

Among them, m_{ij} is the number of occurrences of feature word t_i appearing in the text d_j; n_i is the total number of texts containing feature word t_i; N is the total number of texts in the corpus; F is a normalization factor; $\sum_k m_{k,j}$ is the total number of feature word in the text d_j; and $p(t|\, c_k)$ is the probability that the feature word t appears in the category $c_k \cdot p(c_k\,|t)$ is the conditional probability that the feature belongs to the category c_k when the feature word appears.

Then use the optimized TF-IDF weighting word2vec to train the word vector, assign a weight to each feature word vector, and accumulate each weighted word vector according to the corresponding dimension to obtain the vector representation of each text, that is, updating each text vector according to the formula 4:

$$GW(d) = \sum\nolimits_{t \in d} G(t,d) \;\times\; Word2vec(t) \tag{4}$$

3.4 The Idea of the CD_MTR Model

With the advantages of the TF-IDF weighted vector space model, the LSI model and the CD_STR, they express the text information in different ways. Therefore, the three single-text representation models are combined to allow the three models to complement each other and to better express the content of the text. Thus, a text representation model (CD_MTR) that integrates multiple models is proposed.

The main idea of the CD_MTR model is that each single-text representation model selects an appropriate dimension to vectorize the original text and obtains three different sets of text representation vectors. The union of the text vectors corresponding to these text vector sets is determined as the final text vector. The specific solution is shown in Eq. 5:

$$CD\_MTR(d_i) = LSI(d_i) \oplus T\_VSM(d_i) \oplus CD\_STR(d_i) \tag{5}$$

Among them, d_i is the ith text in the data set D; $\oplus$ is a splice operator; $LSI(d_i)$ is the text vector representation obtained from the LSI model training text d_i; $T\_VSM(d_i)$ is vector representation obtained by the TF-IDF weighted vector space model; and $CD\_STR(d_i)$ is the text vector representation obtained by the CD_STR model training text d_i.

3.5 Structural Improvements for Convolutional Neural Network (MCNN)

In general, only one convolution kernel is included in each convolution layer, and the number of convolution kernels is set to a fixed value. For text data, in order to take the contextual information of each feature word in the text into consideration, a variety of convolution kernels of different sizes can be designed. In this paper, three convolution kernels with different sizes are designed as 3×360, 4×360 and 5×360. The respectively number of corresponding convolution kernels are 150, 100, and 50.

3.6 CD_MTR Combined with Convolutional Neural Network (MTR_MCNN)

For the input features of convolutional neural network, the references [11, 15–17] use the length of the longest text in all the texts of the data set as a benchmark, and the rest of the texts shorter than this length are filled with special characters. For example, if the text is insufficiently long, padding is used to fill it. This method is also a commonly used processing method for input data based on the convolutional neural network text classification method. Two disadvantages are shown in this method:

(1) Taking the length of the longest text in a data set as a benchmark, texts that are shorter than this length are filled with special characters. There will be too many non-semantic characters in short texts, which affects the classification effect.
(2) The text represented by single model is used as the input of the convolutional neural network. The feature representation of the text is relatively single, which is not conducive to text classification.

In order to correct the drawbacks of the above method, a method of combining the CD_MTR model with a convolutional neural network (MTR_MCNN) is proposed. The dimension of each text vector obtained by this method is the same, so it does not need to be filled with special characters. Then, the text retains the original semantic information. And the text vector trained by the CD_MTR model expresses each text in multiple ways as an input to the convolutional neural network, allowing it to extract deeper features and achieve better classification results.

4 Experimental Design

4.1 Experiment Data Set

In order to verify the performance of the CD_MTR model on text classification, two classification data sets were selected for experimentation. A total of 24,000 texts were selected from 6 categories of automotive, culture, economics, medicine, military, and sports of the NetEase News Corpus. There are 7691 texts in 8 categories of Fudan Text Classification Corpus: art, history, computer, environment, agronomy, economics, politics, and sports. The number of corpora categories and the proportionality of texts are not the same. This experiment uses a ten-fold cross validation method to evaluate the effectiveness of this method.

4.2 The Influence of Single-Text Representation Model Dimension on the Effect of CD_MTR Model

The CD_MTR model proposed in this paper combines three single models of T_VSM, LSI and CD_STR. In order to have a good text representation effect for the CD_MTR model, it is necessary to fuse three dimensions of the T_VSM, LSI, and CD_STR. The effect of testing the CD_MTR on two datasets for three single models with different dimensions were chosen (the number of topics for the LSI is 400). This paper tests the

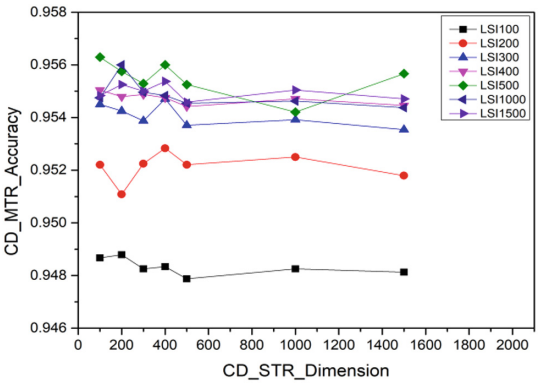

Fig. 2. The effect of LSI and CD_STR dimensions on CD_MTR classification effect when T_VSM dimension is 100

various combinations of the three single models of T_VSM, LSI and CD_STR when dimensions of [100, 200, 300, 400, 500, 1000, 1500] are selected. The combination is too much. This paper only takes part of them to explain. Respectively shown in Figs. 2, 3, and 4.

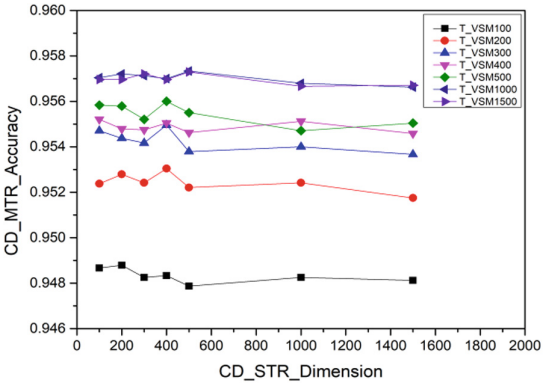

Fig. 3. The effect of T_VSM and CD_STR dimensions on the CD_MTR classification effect when the LSI dimension is 100

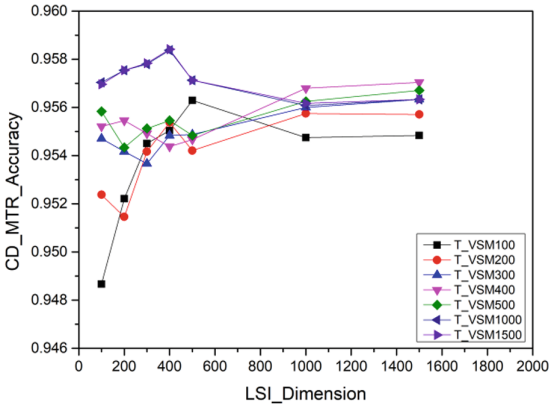

Fig. 4. The effect of T_VSM and LSI dimensions on CD_MTR classification effect when CD_STR dimension is 100

The results in Figs. 2, 3 and 4 show that changes in the three single-model dimensions of T_VSM, LSI, and CD_STR affect the text representation capability of the fusion model CD_MTR. Considering the classification effect and classification speed of the four models of T_VSM, LSI, CD_STR and CD_MTR in different dimensions, the dimensions of the three models T_VSM, LSI and CD_STR are

selected to be 1000, 500 and 400 respectively. The number of LSI model topics was selected as 400, so the dimension of the CD_MTR model is 1800.

4.3 Text Representation Model Comparison and Analysis

In order to verify the validity of the CD_MTR model, we compared the effect of different models: A_word2vec (an average of each word vector per text), T_word2vec (TF-IDF+word2vec), CD_STR, LDA fusion word2vec (LDA+word2vec), T_VSM fusion LSI (T_VSM+LSI), LSI fusion CD_STR (LSI+CD_STR), and T_VSM fusion CD_STR (T_VSM+CD_STR). The classification effect of each model is shown in Table 1.

Table 1. Classification effect of each model on two datasets

Methods	NetEase news text (%)		Fudan text (%)	
	Micro-average F_1	Macro-average F_1	Micro-average F_1	Macro-average F_1
A_word2vec	91.79	91.83	92.20	90.59
T_word2vec	93.24	93.25	91.92	90.18
CD_STR	94.24	94.25	93.08	92.39
LDA+word2vec	92.99	93.00	93.80	93.01
T_VSM+LSI	94.84	94.85	95.97	95.66
LSI+CD_STR	95.58	95.59	96.78	96.49
T_VSM +CD_STR	95.70	95.70	96.76	96.44
CD_MTR	95.85	95.86	96.93	96.56

The results from Table 1 show that:

(1) The micro-average F1 value and the macro-average F1 value obtained by CD_STR on the two data sets are superior to the single models A_word2vec and T_word2vec. This result also verifies that the CD_STR model considers the influence of a single word on the entire document and has better class discrimination ability.
(2) Compared with other combined models, the CD_MTR model presented in this paper improves both the micro-average F1 value and the macro-average F1 value.

4.4 Ten-Fold Cross Result

In order to test the effectiveness of the MTR_MCNN method proposed in this chapter, a ten-fold cross validation was used. The ten-fold cross-validation method divides the data set into 10 equal and disjoint sub-samples each time. In 10 sub-samples, one sub-sample is used as the data of the test model, and the other 9 samples are used for training. Verifying each sub-sample for one time and repeat cross validation for 10

times. Figure 5 show the classification effect of the NetEase news text and Fudan text under different sub-samples, respectively.

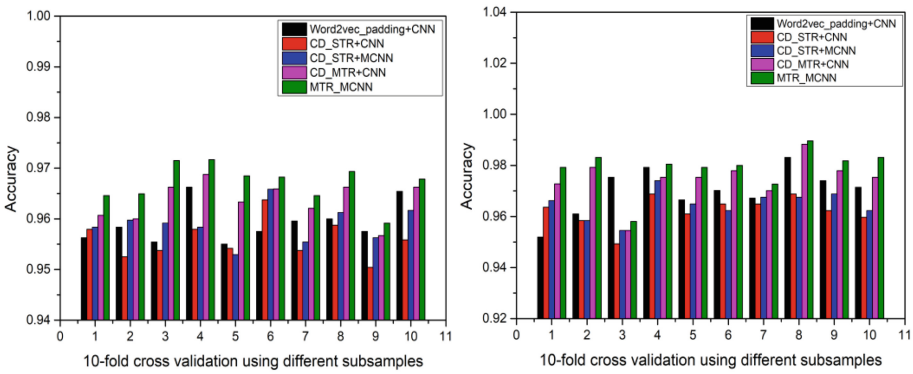

Fig. 5. Different methods of ten-fold cross-validation

The results in Fig. 5 show that:

(1) Compared with CD_STR+CNN, CD_STR+MCNN in the distribution of ten different training sets/test sets pairs is better in most cases. MTR_MCNN is superior to CD_MTR+CNN.
(2) The classification accuracy of CD_MTR+CNN is better than CD_STR+CNN, and the classification accuracy of MTR_MCNN is also superior to CD_STR+MCNN. Furthermore, under the same convolutional neural network structure, the fusion model CD_MTR presented in this paper can better represent text information, distinguish categories, and improve the accuracy of text classification.
(3) The model of word2vec_padding+CNN performs better under certain training set/test set pairs in normal conditions. But compared with the MTR_MCNN presented in this chapter, the classification accuracy is lower than that of MTR_MCNN. In the experiment, the word2vec_padding+CNN method needs to fill in most of the texts in the text sets with special characters, and there is a case where the supplemented text information and the original text information are deviated which affects the classification effect. In this method, the matrix dimension of the input convolutional neural network is the number of words per text multiplied by the dimension of each feature word. When the number of text feature words is too large, the text matrix dimension of the input convolutional neural network will be very large, thus affecting the training speed.

4.5 Method Comparison and Analysis

To further verify the validity of the MTR_MCNN method, this paper compares it with other classification methods. Table 2 shows the average classification accuracy of each method under the 10-fold crossover method.

Table 2. Classification accuracy of different text classification methods (%)

Methods	NetEase news text	Fudan text
	Accuracy (%)	Accuracy (%)
CD_MTR	95.85	96.93
CD_STR+CNN	95.55	96.21
CD_STR+MCNN	95.88	96.46
Word2vec_padding+CNN	95.91	96.99
CD_MTR+CNN	96.36	97.46
MTR_MCNN	96.70	97.87

From the results in Table 2, we can conclude:

(1) The text vector represented by the single model CD_STR is lower in classification accuracy than the CD_MTR proposed in Sect. 3.4 of this paper, whether it is an input as a non-optimized or optimized convolutional neural network. The classification accuracy of the method once again proves the performance of the CD_MTR model.
(2) The MTR_MCNN method proposed in this paper has the highest classification accuracy among all the methods. The accuracy values on the two data sets respectively are 96.70% and 97.87%, and its classification is also more effective than the common used word2vec_padding+CNN method.

The above results are mainly because the MTR_MCNN method proposed in this paper introduces a convolutional neural network to improve the feature extraction of the CD_MTR method which is superficial. The MTR_MCNN method designs different convolution kernels of different sizes and numbers, which extracts text features from different angles. The MTR_MCNN method uses the CD_MTR model to vectorize the text and convert the resulting text vectors into a matrix form as input to convolutional neural network. Additionally, the MTR_MCNN method do not need to fill shorter texts with special characters, which affects the expression of the original text information, so it improved the text classification accuracy.

5 Conclusion

This paper proposes a single model CD_STR and a fusion model CD_MTR, which using optimized TF-IDF weighting word2vec with semantic information combines with LSI and TF-IDF weighted vector space model to complement each other. Based on the CD_MTR model, the classification method MTR_MCNN combined with CD_MTR and convolutional neural network is proposed in this paper. In this method, the convolutional neural network structure is improved, and different sizes and numbers of convolution kernels are designed to extract text features from different angles. In addition, the text vectors obtained from the CD_MTR are converted into a matrix form as an input to the convolutional neural network. For MTR_MCNN method, it does not need special characters to fill the text, avoiding meaningless additional text information

and reducing the dimension of the input matrix so as to improve the training speed of the convolutional neural network. The experiment results show that both the CD_STR model and CD_MTR model and the MTR_MCNN method in this paper have achieved good classification effect and are superior to other methods.

References

1. Hinton, G.E.: Learning distributed representations of concepts. In: Eighth Conference of the Cognitive Science Society, pp. 1–12 (1986)
2. Bengio, Y., Ducharme, R., Vincent, P., et al.: A neural probabilistic language model. J. Mach. Learn. Res. **3**(2), 1137–1155 (2003)
3. Mnih, A., Hinton, G.: A scalable hierarchical distributed language model. In: International Conference on Neural Information Processing Systems, pp. 1081–1088. Curran Associates Inc., (2008)
4. Mikolov, T., Chen, K., Corrado, G., et al.: Efficient estimation of word representations in vector space. Comput. Sci. (2013)
5. Mikolov, T., Yih, W.T., Zweig, G.: Linguistic regularities in continuous space word representations. In: HLT-NAACL (2013)
6. Hu, B., Lu, Z., Li, H., et al.: Convolutional neural network architectures for matching natural language sentences. In: International Conference on Neural Information Processing Systems, pp. 2042–2050. MIT Press (2014)
7. Yan, Y.: Text representation and classification with deep learning. University of Science and Technology, Beijing (2016)
8. Salton, G.: A vector space model for automatic indexing. Commun. ACM **18**(11), 613–620 (1975)
9. Lecun, Y., Boser, B., Denker, J.S., et al.: Backpropagation applied to handwritten zip code recognition. Neural Comput. **1**(4), 541–551 (2014)
10. Bouvrie, J.: Notes on convolutional neural network. Neural Nets (2006)
11. Liu, X., Zhang, Y., Zheng, Q.: Sentiment classification of short texts on internet based on convolutional neural network model. Comput. Mod. **2017**(4), 73–77 (2017)
12. Wang, P., Xu, J., Xu, B., et al.: Semantic clustering and convolutional neural network for short text categorization. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, vol. 2, pp. 352–357 (2015)
13. Zhou, C., Sun, C., Liu, Z., et al.: A C-LSTM neural network for text classification. Comput. Sci. **1**(4), 39–44 (2015)
14. Lai, S., Xu, L.H., Liu, K., et al.: Recurrent convolutional neural networks for text classification. In: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, pp. 2267–2273 (2015)
15. Cai, H.: Research of short-text classification method based on convolution neural network. Southwest University (2016)
16. Yin, Y., Yang, W., Yang, H., et al.: Research on short text classification algorithm based on convolutional neural network and KNN. Comput. Eng. (2017)
17. Kim, Y.: Convolutional Neural network for Sentence Classification. Eprint Arxiv (2014)