Information Fusion 64 (2020) 1–11

Contents lists available at ScienceDirect

Information Fusion

journal homepage: www.elsevier.com/locate/inffus

Full length article

Deep learning based emotion analysis of microblog texts

Dongliang Xu[a,b], Zhihong Tian[a,*], Rufeng Lai[c], Xiangtao Kong[b], Zhiyuan Tan[d], Wei Shi[e]

[a] Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou 510006, PR China
[b] School of Mechanical Electrical and Information Engineering, Shandong University, Weihai 264209, PR China
[c] Beijing Key Laboratory of Digital Media, School of Computer Science and Engineering, Beihang University, Beijing 100191, PR China
[d] School of Computing, Edinburgh Napier University, Edinburgh, UK
[e] School of Information Technology, Carleton University, Ottawa, Canada

ARTICLE INFO

Keywords:
Microblog short text
Emotional analysis
Convolutional neural network
Word2vec

ABSTRACT

Traditional text emotion analysis methods are primarily devoted to studying extended texts, such as news reports and full-length documents. Microblogs are considered short texts that are often characterized by large noises, new words, and abbreviations. Previous emotion classification methods usually fail to extract significant features and achieve poor classification effect when applied to processing of short texts or micro-texts. This study proposes a microblog emotion classification model, namely, CNN_Text_Word2vec, on the basis of convolutional neural network (CNN) to solve the above-mentioned problems. CNN_Text_Word2vec introduces a word2vec neural network model to train distributed word embeddings on every single word. The trained word vectors are used as input features for the model to learn microblog text features through parallel convolution layers with multiple convolution kernels of different sizes. Experiment results show that the overall accuracy rate of CNN_Text_Word2vec is 7.0% higher than that achieved by current mainstream methods, such as SVM, LSTM and RNN. Moreover, this study explores the impact of different semantic units on the accuracy of CNN_Text_Word2vec, specifically in processing of Chinese texts. The experimental results show that comparing to using feature vectors obtained from training words, feature vector obtained from training Chinese characters yields a better performance.

1. Introduction

Microblog has become a popular channel for people to communicate and express emotions. The massive amount of microblogs generated daily provides a favorable data base for text emotion analysis [1,2]. Micro-texts have a more complex emotional vocabulary than commentary texts, such as news, movies, and documents. Therefore, conducting emotional analysis on microblog texts is a challenging task that has received considerable research attention [3,4].

Emotion classification (EC) is an important method for automatic mining and analysis of subjective information, such as views, opinions, emotions, and likes and dislikes in texts. EC identifies the sentimental polarities (positive or negative) of a given text and then classifies the text accordingly. Most existing emotion analysis methods are either dictionary-based rule methods [5–10] or statistical machine learning methods [11–14]. A dictionary-based rule method computes the emotional tendency of a microblog in accordance with an emotional dictionary that consists of a list of predefined emotion vocabularies. A machine learning-based method (shallow learning) considers emotional analysis as a problem of pattern classification and builds a classification model to predict the sentimental polarity of a microblog.

Most traditional emotion classification methods are designed for processing of extended texts. When it comes to processing of micro-texts such as microblogs, these methods fail to achieve good classification performance because the extracted features are inconspicuous. Moreover, these traditional methods do not take into consideration the semantic relevance between features. For example, if the word ”suffering” appears in the text to be classified and the only word used in a positively-labeled text is ”sad”, then accurate classification cannot be performed.

In this paper, we propose a microblog emotion classification model, CNN_Text_Word2vec, intending to solve the above-mentioned problems using a convolutional neural network (CNN). The main contributions of this paper are as follows:

1. In view of the characteristics of Chinese texts, the word-2vec neural network model is proposed to classify the emotions in Chinese microblog texts based on the feature vectors of single characters. The experimental results show that the feature vectors composed by Chinese characters are more helpful for emotional classification of Chinese microblog short texts than those composed by Chinese words.
2. The feature vectors trained by the word2vec model are proposed as input features. The convolutional neural network model com-

* Corresponding author.
E-mail address: tianzhihong@gzhu.edu.cn (Z. Tian).

https://doi.org/10.1016/j.inffus.2020.06.002
Received 2 September 2019; Received in revised form 28 May 2020; Accepted 13 June 2020
Available online 20 June 2020

bined with the attention mechanism with multiple convolutional kernel of different sizes is established to realize end-to-end integrated training and improve the accuracy of the text emotion classification.

3. Comparative evaluations on the overall accuracy of emotional classification in the following four models have been conducted and reported in this paper: CNN_-Text_word2vec (with and without word2vec word vector used), SVM, RNN, LSTM. The experiment results show that the emotional classification accuracy of CN-N_Text_word2vec on Chinese microblogs are 7%, 6.9% and 2.91% higher than that of SVM, RNN, LSTM respectively. Furthermore, we also conduct experiments and report the results on the overall accuracies of CNN-_Text_word2vec when semantic units (i.e. both word vectors and character vectors) of different scales are used. Comparative evaluation results demonstrate a consistently higher accuracy when character vectors are used than word vectors in each of the scale setting tested.

2. Related work

2.1. Traditional emotional classification approaches

By learning methods, traditional emotion classification methods can be divided into three categories, namely, supervised, unsupervised, and semi-supervised learning methods.

Supervised learning methods mainly use machine learning methods for emotion classification. Pang et al. [15] used online emotional analysis to study online movie reviews and compared the performance of naive Bayes, maximum entropy classification, and support vector machines (SVMs). The results showed that SVMs achieved a higher accuracy than other methods. Mullen et al. [16] used SVMs to synthesize various feature information from different sources using a unigram model and achieved better classification performance than the work of Pang et al. Kennedy et al. [17], through combined use of contextual valence shifters and SVMs, achieved favorable results in emotion classification. Abbasi et al. [18] applied the emotion classification method to English and Arabic websites and used the entropy-weighted genetic algorithm (EWGA) to improve classification accuracy.

Unsupervised learning methods mainly classify emotions based on prior knowledge. These methods are less effective than their supervised counterparts but have attracted wide attention from researchers because they do not entail a large labeled corpus. Turney [19] determined whether a review was recommended or not by identifying the average semantic orientation of the phrases that contained adjectives or adverbs in the review. The semantic orientation of a phrase was obtained on the basis of the mutual information between a given phrase and the word ”excellent” minus the mutual information between the given phrase and the word ”poor”. Wilson et al. [20] manually labeled a group of words using a manual annotator to construct an emotional dictionary. The system they built automatically recognized the context polarity of a subset of emotional expressions and achieved favorable results. Adreevskaia et al. [21] used a dictionary (WordNet) to learn the emotional direction of a text based on semantically relevant words that were mined in the dictionary. Lu et al. [22] used a corpus-based method to infer the emotional orientation of words in a given corpus through analysis of the relationship between words and certain observed information; moreover, they built an emotional language vocabulary for a specific domain. Zagibalov et al. [23] used an automatic seed word selection method for unsupervised emotion classification of Chinese product reviews. This method did not require any labeled training data and achieved 92% F1.

The semi-supervised learning method is a suitable option when there are only few labeled datasets. It consumes less time and manpower than supervised learning methods and achieves better classification effect than unsupervised learning methods. Therefore, semi-supervised learning methods demonstrate advantages over the other two types of methods. Sindhwani et al. [24] proposed a method that incorporated labeled features and unlabeled documents into a standard regularized least squares method. In scenarios where labeled data were limited and unlabeled data were the majority, this method achieved better results than the pure supervised methods and the competitive semi-supervised learning methods. Dasgupta and Ng [25] first used spectral techniques to mine explicit unambiguous reviews and used them to classify ambiguous reviews through a combination of active, transductive, and ensemble learning. They combined various semi-supervised emotion classification methods and achieved favorable classification results. Wan [26] applied a collaborative training method to the semi-supervised learning with labeled English corpus and unlabeled Chinese corpus for Chinese emotion classification, which solved the problem of cross-language emotion classification. The semi-super-vised emotion classification model proposed by Li et al. [27] can effectively solve the problem of non-equilibrium emotion classification.

2.2. Emotion classification based on deep learning

Deep learning models have achieved remarkable results in computer vision [28] and speech recognition [29] in recent years. It has also been applied to natural language processing tasks, including the study of word embedding and training of texts. Bengio et al. [30] proposed an architecture for estimating neural network language models that used a feedforward neural network with a linear projection and a nonlinear hidden layer to learn word embedding representation and statistical language models simultaneously. Their work has been cited in many studies. Mikolov et al. [31,32] introduced Skip-Gram, an effective means of learning high-quality distributed vector representations. Skip-Gram can learn to represent fixed-length embedding vectors by predicting adjacent terms in a document. Glorot et al. [33] proposed a deep learning method for the domain adaptation problem of emotion classifiers and applied this method to emotion classification of large-scale online reviews in order to improve classification accuracy. Socher et al. [34] introduced a recursive neural network that represented phrases as word embeddings and analysis trees to realize emotion classification. Functions on the basis of a correction factor is then used to calculate the word embedding.

Most existing algorithms that learn continuous word representations typically only model the syntactic context of words but ignore the sentiment of the text. Tang et al.[35] addressed this issue by learning sentiment specific word embeddings(SSWEs), which encodes sentiment information in the continuous representation of words. Ren et al.[36] proposed a context-based neural network model for Twitter sentiment analysis that incorporated contextualized features from relevant tweets into the model in the form of word embeddings.

A CNN uses convolution filters to learn local features [37–39]. First used in computer vision, CNNs have then been applied to natural language processing and achieved favorable results in semantic analysis [40], search query retrieval [41], sentence modeling [42], and other conventional natural language processing (NLP) tasks [43,44].

Application of CNNs in text classification has attracted increasing research attention. Satapathy [45] is the first of its kind to incorporate deep learning into a microtext normalization module and improve the sentiment analysis task. Wang et al.[46] proposed a jointed CNN and RNN architecture, taking advantage of the coarse-grained local features generated by the CNN and long-distance dependency learned via the RNN for sentiment analysis of short texts. Arora [47] proposes a text normalization with deep convolutional character level embedding (Conv-char-Emb) neural network model for SA of unstructured data. Kim [48] used a CNN to classify sentences pre-processed by word embeddings and built a text classification model on the basis of the CNN. The experiment results showed that the CNN-based text classification method achieved higher accuracy than the optimal method did. By drawing inspiration from the work by Kim et al., we built a model that used multiple convolution kernels of different sizes to learn the feature vectors at the sentence level, concatenated the feature vectors and con-

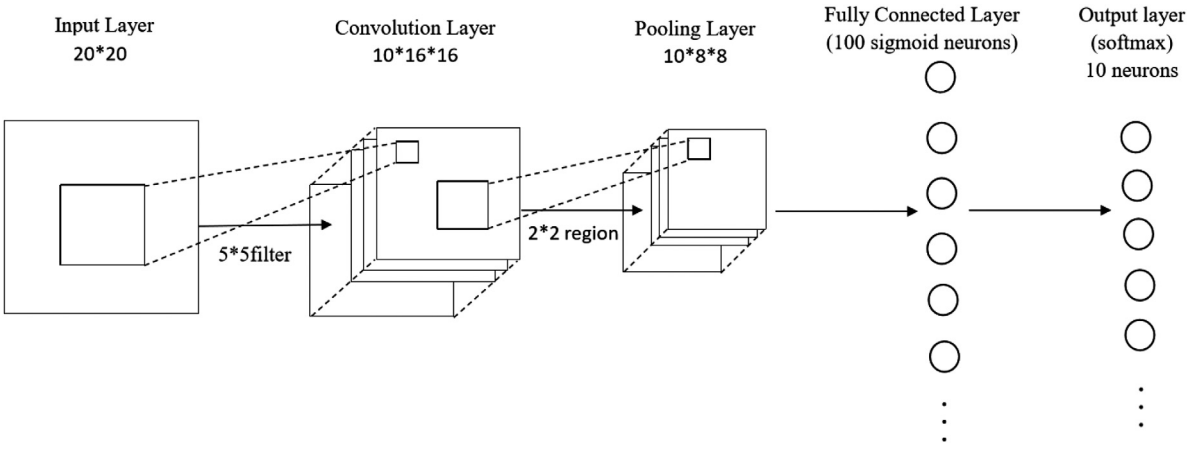


Fig. 1. Typical CNN model.

structed new sentence feature vectors for the emotion classification on microblogs.

3. Research background

3.1. CNN

Traditional feedforward neural networks connect each input neuron to its corresponding output neuron on the following layer. This process is called a full connection. However, this method entails additional calculation of weights and offsets, which seriously affects the training speed. Instead of adopting full connection, CNN uses partial connection, that is, each neuron is linked to only a region of the input layer which is referred to as a local receptive field of the hidden neuron. Another feature of CNNs is the use of shared weights. A neuron in a hidden layer is the result of convolution of hidden neurons from a local receptive field. A local receptive field generates one neuron for the next layer each time it moves. In each convolution kernel, the weight used, including the size and the value, remains unchanged. Thus, the number of training in a CNN does not depend on the number of neurons, but on the size of the convolution kernel. Therefore, a CNN has fewer parameters and a faster training speed than previous neural network models.

A typical CNN model consists of the following layers: input, convolution, pooling, fully-connected, and output layers. Fig. 1 illustrates the structure of a CNN.

In this example, the input layer has 20X20 neurons and 10 feature maps from the input layer to the convolution layer. Each feature map defines a 5X5 shared weight and a single shared offset, which defines a 5X5 partial receptive field. Then, the partial receptive field will move across the entire image. For each partial receptive field, a hidden layer will have a different hidden neuron. Thus, each feature map will have a 16X16 neuron output from the convolution layer; for all feature maps, the result is a 10X16X16 hidden-feature neuron layer. Next, the pooling layer is applied to a 2X2 region, which is considered max-pooling of the eigenvalues of the 10 feature maps. The result is a 10X8X8 hidden-feature neuron layer. The next layer in the network is a fully-connected layer that connects every neuron in the pooling layer to every neuron in the output layer. Finally, an output layer of softmax is used, and the input and output samples belong to the probability of each class.

3.2. Word2Vec model

Word2Vec is a neural network model proposed by Mikolov et al. [31,32,49–51] for training of distributed word embedding representations, including CBOW and Skip-Gram. The former trains the current word embedding by the context, whereas the latter predicts the context according to the current word. The word embedding trained by a

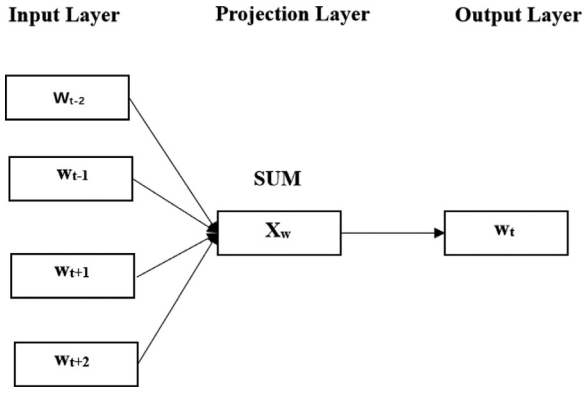


Fig. 2. CBOW neural network model.

Word2Vec model captures semantic similarity between words and fully considers the semantic information of words. In our study, a CBOW model based on negative sampling is used to train the word embedding. The CBOW model is briefly introduced as follows.

In Fig. 2, the CBOW neural network model consists of three layers, namely, input, projection, and output. A given corpus C is assumed, with its word sequence as $(w_{t-2}, w_{t-1}, w_t, w_{t+1}, w_{t+2})$, where w_t is the current word and the remaining words are the context of w_t. The input layer is the word embedding that corresponds to the two words before and after the word w_t, that is, the word embedding of word w_t: Context(w_t); the projection layer is the accumulation of the five-word embedding of the input layer, where X_w is obtained. Random negative sampling is performed at the output layer to predict w. The negative sampling method assumes that, for a given Context(w), the word w is a positive sample, and the other words are negative. Therefore, an arbitrary word v is expressed as:

$$L^w(v) = \begin{cases} 1, v = w \\ 0, v \neq w \end{cases}, \tag{1}$$

where $L^w(v)$ represents the label of word v; that is, 1 represents a positive sample, and 0 represents a negative sample. For a given positive sample (Context(w), w), the final goal is to maximize G:

$$G = \log \prod_{w \in C} g(w), \tag{2}$$

$$g(w) = \prod_{u} p(u|Context(w)). \tag{3}$$

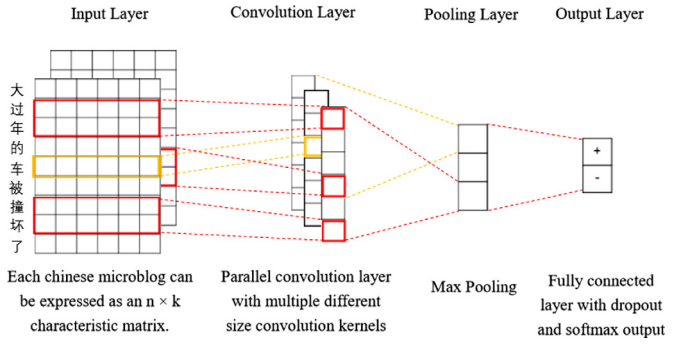


Fig. 3. CNN_Text_Word2vec model structure.

The arbitrary word u corresponds to an assistant vector θ^u, and Function p is defined as

$$p(u|Context(w)) = \begin{cases} \sigma(X_w^T\theta^u), L^w(u) = 1 \\ 1 - \sigma(X_w^T\theta^u), L^w(u) \neq 0 \end{cases}, \quad (4)$$

where $\sigma(x)$ is the sigmoid function, and Function p is expanded to

$$p(u|Context(w)) = [\sigma(X_w^T\theta^u)]^{L^w(u)}[1 - \sigma(X_w^T\theta^u)]^{1-L^w(u)}, \quad (5)$$

Substitution of Formula (5) into Formula (3) results in

$$g(w) = \sigma(X_w^T\theta^u) \prod_{u,u\neq w} [1 - \sigma(X_w^T\theta^u)]. \quad (6)$$

Therefore, maximizing G, that is, maximizing $g(w)$, is equivalent to maximizing $\sigma(X_w^T\theta^w)$ while minimizing $\sigma(X_w^T\theta^u)$. In particular, the probability of a negative sample decreases while that of a positive sample increases. Consequently, this is a model that we will adapt to solve the problems of existing emotion classification models.

4. CNN_Text_Word2vec

The CNN_Text_Word2vec is constructed on the basis of a CNN model. Word2vec is introduced in the input layer to pre-train the word embeddings. In the convolution layer, multiple convolution kernels of different sizes are used to learn microblog text features in parallel. Maximum pooling is performed to generalize features in the pooling layer, and the classification result is generated in the output layer. Fig. 3 presents the structure of the model.

4.1. Input layer

In order to fully utilize the characteristic of words, during the preprocessing step, each microblog in the dataset is divided into multiple Chinese characters (referred to as words hereafter) separated by spaces, and a word embedding is trained on every single character. Word2Vec is used when each word embedding is trained. The word embedding of a specific word is obtained by specification of corresponding parameters, including the dimensions of word embedding, the number of iterations, and the size of the context window (i.e. the number of characters in each window). Each word that appears more than once is added to a dictionary of mXk dimensions, where m is the number of words in the dictionary, and k is the dimension of the word embedding specified during training.

The input for Microblog x that includes n characters is expressed as:

$$x_{1:n} = x_1 \oplus x_2 \oplus \ldots \oplus x_n, \quad (7)$$

where $\oplus$ is a joint operator. $x_i \in R^k$ is the k-dimensional word embedding that corresponds to the i words in Microblog x. Therefore, Microblog x can be expressed as a feature matrix of nXk, where n is the length of the microblog, that is, the number of Chinese characters in the microblog; k is the dimension of the word embedding specified in the previous training, that is, each Chinese character is expressed as a k-dimensional word embedding. Then, the feature matrix of Microblog x is used as the input of the CNN, and feature extraction is performed by the convolution and pooling layers of the CNN to obtain sentence expressions of the microblog.

4.2. Convolution layer

Our study uses the parallel convolution layer with multiple convolution kernels of different sizes to learn microblog text features. Multiple convolution kernels are set to acquire features in the microblog sentence expression comprehensively and reduce the degree of fortuity in the feature extraction process. For example, we can set the filter size of convolution kernels as h_1Xk, h_2Xk, h_3Xk, where k is an integer as well as the dimension of word embeddings, and h_i is the stride value (i.e. the number of words that a sliding window slides over each time it moves). For a convolution kernel, a feature c_i is calculated using the following equation:

$$c_i = f(w \cdot x_{i:i+h-1} + b), \quad (8)$$

where $w \in R^{hk}$ is the shared weight, $x_{i:i+h-1}$ represents the connection of the word embedding, that is from the i word of microblog x to the $i + h - 1$ word, ordered from top to bottom. $b \in R$ is an offset term, and f is a nonlinear function, such as a hyperbolic tangent function ($\tan h$) and a rectified linear unit (ReLU). The ReLU is used in our study. Therefore,

$$c_i = max(0, w \cdot x_{i:i+h-1} + b). \quad (9)$$

For this convolution kernel, the convolution of h words is conducted successively, and the length of the microblog is n. Thus, $n - h + 1$ convolutions are performed, resulting in the following feature map:

$$c = [c_1, c_2, \ldots, c_{n-h+1}] \quad (10)$$

where $c \in R^{n-h+1}$. Therefore, $n - h + 1$ feature maps are used for each convolution kernel to obtain a feature vector t whose dimension is 1 X $(n - h + 1)$. If the number of convolution kernels is p, then p feature vectors can be obtained through feature mapping, and $T = [t_1, t_2, \ldots, t_p]$.

If q parallel convolution kernels of different types (i.e. of various sizes: h_1Xk, h_2X$k, \ldots, h_q$Xk) are used and the number of each type of convolution kernel is p, then (pXq) feature vectors can be obtained after feature mapping, and $S = [t_1, t_2, \ldots, t_p]$. Here, S is the output from the convolution layer and then passed to the pooling layer of the CNN.

4.3. Pooling layer

In the pooling layer, the maximum pooling of the non-linear downsampling (max-overtime pooling) is applied to the 1 X $(n - h + 1)$ region on the feature map, and the maximum value ($\hat{c} = max\{c\}$) is used as the feature that corresponds to the feature map [52,53]. That is, the largest value is extracted from the previous feature map to represent the most important signal. Therefore, for the feature vector t obtained for each convolution kernel, the eigenvalue (v) with a dimension of 1 is obtained after pooling. This pooling method can provide the input of varied-length sentences because if the maximum value is extracted after pooling, then a fixed-length sentence feature vector, namely, the eigenvalue (1) of dimension v, can be obtained, though the length of each microblog in the microblog dataset is different.

The process of extracting a feature from a convolution kernel is described above. The model uses multiple convolution kernels to obtain multiple features, such as filtering three, four, or five words separately. If q parallel convolution kernels of different sizes are used in the convolution layer and the number of each convolution kernel is p, then $p \cdot q$ 1-D eigenvalues can be obtained after the convolution and pooling operations. Finally, all pooled features are combined to obtain a feature vector V with a dimension of 1 X $p \cdot q$. Therefore, the feature vector V of the microblog is obtained and transmitted to the output layer.

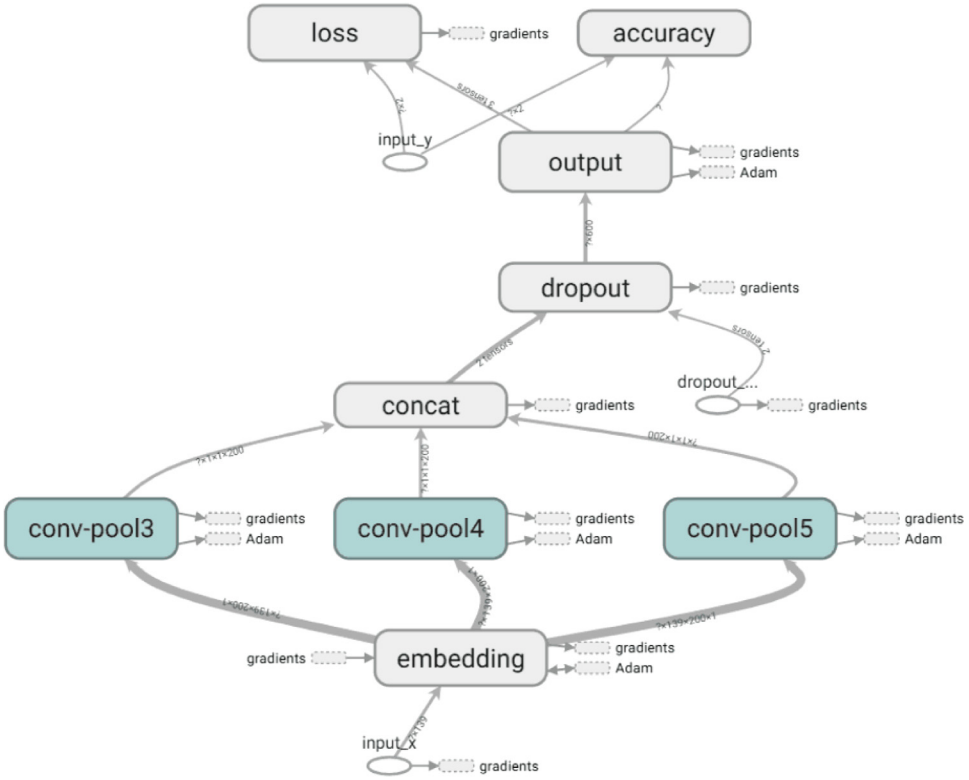


Fig. 4. Visual network diagram in TensorBoard.

4.4. Output layer

The output of the pooling layer vector is connected to the softmax layer through full connection. Therefore, the last layer is the fully-connected softmax layer for classification whose output is the probability distribution of the final category. In the final implementation, the dropout technique is used on the fully-connected layer to prevent the hidden layer neurons from self-adapting and to reduce overfitting, and the L2 regularization limit [7] is provided to the weight parameters on the fully-connected layer. Therefore,

$$z = w \cdot (V \circ r) + b, \tag{11}$$

where $\circ$ means multiplying corresponding matrix elements. $r \in R^{pXq}$ is a Bernoulli variance. If softmax output probability distribution is used, then the activation value a_j of the j neuron is

$$a_j = \frac{e^{z_j}}{\sum_k e^{z_k}}, \tag{12}$$

where the output activation value adds up to 1, that is,

$$\sum_j a_j = \frac{\sum_j e^{z_j}}{\sum_k e^{z_k}} = 1. \tag{13}$$

On the basis of the Chinese microblog corpus that has been crawled, this study adopts binary classification with either positive or negative as a classifications result. Therefore, the model solves the binary classification problem of the emotional polarity of a microblog, which is classified as formula (14), μ is a superparameter.

$$classify(V) = \begin{cases} 0, a_0 \geq \mu \\ 1, a_0 < \mu \end{cases}. \tag{14}$$

Thus, the model converts each microblog into sentence expressions of the feature space in the input layer, performs feature extraction and generalization through CNN convolution and pooling operations, and realizes emotion classification of the microblog in the output layer.

Python is used to implement and visualize the neural network model on TensorBoard, as depicted in Fig. 4. In this figure, conv-maxpool-3, conv-maxpool-4 and conv-maxpool-5 are ommitted due to space limitation. In the network, the "embedding layer" represents the input layer, and the conv-maxpool represents the convolution and pooling layers. There are three parallel convolution layers, each of which is followed and connected to a pooling layer. The eigenvalues obtained by three-layer convolution and pooling are concatenated by a concat and passed to the output layer. The output layer is a fully connected softmax layer with the dropout technique. The output layer outputs the classification accuracy and loss of the model.

5. Experiments and results analysis

Four experiments have been conducted to verify validity of the proposed model. The experiment setting is introduced in the following part. Four groups of comparative experiments are presented, and the experiment results are analyzed.

5.1. Experiment setup

5.1.1. Emotional analysis dataset

In this study, we crawled approximately 100,000 Chinese microblogs from Sina Weibo. A total of 80,000 microblogs were retained, 40,000 of which were positive and 40,000 negative, after de-duplication, screening, and manual labeling. The microblogs were divided into the training and testing sets. The training set has 60,000 microblogs, 30,000 positive and 30,000 negative. The testing set has 20,000 microblogs, 10,000 positive and 10,000 negative. Abbreviations and spelling errors were considered legitimate and added to the trained dataset after labling. A sample of the dataset is summarized in Table 1.

5.1.2. Parameters

The microblog text was pre-processed. First, the characters other than Chinese in a microblog text were converted into a space, and then multiple consecutive spaces were converted into a space. The space characters before and after the sentence were removed, and each word was separated by a space.

Table 1
Sample Microblog Dataset .

	Positive sample	Negative sample
1	It's fun, the Spring Festival is more fulfilling. [very happy]	The car was hit on the Spring Festival! [hum] so annoying [cursing angrily] my lovely car
2	I think... this guy's listening comprehension is impressive... [lol][lol]	Fortunately, I don't take English listening exam... It makes me dizzy! [mad]
3	Dinner is ready! [hug] What is this dish? Quiz quiz!!	I didn't have dinner, now I am prepared to sleep in this heavy rain [tears]
4	Cracking seeds and eating snacks in a cool breeze. This is the way to enjoy the summer! [hehe][breeze]	What the hell, it's so windy, my car is parked but rocked by the wind [tears][tears][tears] It's really horrible, try not to go outside if you can

Table 2
Default Values of the CNN_Text_Word2vec Model Hyper Parameters .

Parameters	Parameter name	Value
η	Gradient descent learning rate	0.001
pooling	Pooling operator	1-max pooling
filterSize	The size of convolution kernel (window size)	3,4,5
dropout_ keep_prob	Dropout probability (training set)	0.5
batch_size	Number of mini-batch samples	50

When a word did not appear in a dictionary for the first time, it is added to the dictionary after being labelled.

Second, each character in a microblog was trained for word embeddings by the word2vec tool. When a word did not appear in a dictionary for the first time, it would be added to the dictionary after being labelled. Finally, a dictionary containing 4,205 items was obtained. The word vector dimension during the training process was set to 200, and the CBOW algorithm was selected; the iteration times (iter) was set to 5, and the context window was 8.

1. CNN hyper parameter settings
   Experiments were conducted on multiple CNN models.
   The Adam optimization algorithm [56] was used in the training process to perform a stochastic gradient descent (SGD) on out-of-order mini-batch samples. The row vector dimension of the convolution kernel is consistent with the word vector dimension of the word2vec training. Other parameters of the model are listed in Table 2.

5.1.3. Experiment evaluation criteria

Four comparative experiments were set up in this paper. The evaluation indices of the experiment included accuracy (*P*), recall (*R*), positive and negative class F1 values.

For the overall performance, the overall correction rate *Accuracy* is used, and the calculation formula is:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}. \tag{15}$$

In this formula, *TP, FP, TN*, and *FN* represent the correctly-classified positive microblogs, the mis-classified positive microblogs, the correctly-classified negative microblogs, and the mis-classified negative microblogs, respectively.

5.1.4. Comparative evaluation

Four sets of experiments were performed in order to evaluate our proposed model against other existing models. The first set of experiments (E1) compared the emotion classification accuracy of different models. The second set (E2) compared the accuracy of emotion classification of a model using pre-processed word2vec word vectors and that of a model using randomly initialized word embeddings. The third set (E3) compared the accuracy of models under different semantic units. The fourth set (E4) compared the accuracy of the models under different parameters.

In the following sets of experiments, the training set was divided into a training and a verification set. A 10-fold cross-validation method was adopted. That is, the training dataset was divided into 10 mutually exclusive subsets, first of which was used as the verification set, and the remaining nine were used as the training set. The training set contained 27,000 microblogs, and the verification set contained 3,000. 10 training and verification tests were performed to obtain 10 models. The average of these 10 test results was used as the final result.

5.2. Experiment results

5.2.1. Experiment 1

The first set of experiments compared the CNN_Text_-Word2vec model with traditional classification methods, including SVM, logistic regression, random forest, decision tree, and naive Bayes. Their parameters setup is as follows: the parameters *c* and *g* of SVM are got by grid search. The parameter *n_estimators* of Random Forest is 50. The parameters setup of LR, Decision Tree, Naïve Bayes are default parameters setup of sklearn. RNN is MV-RNN of reference [54], which of their parameters setup is the same. LSTM is Tree-LSTM of reference [55], which of their parameters setup is the same. The results are displayed in Table 3 and Fig. 5.

The experiment results showed that CNN_Text_Word2vec had better performance (higher overall accuracy, two types of F1 values) than the traditional classification methods. The naive Bayes algorithm had low overall performance indices with an overall accuracy rate of 80.4%, and the two types of F1 values were 80.74% and 79.95%. The overall performance indices of Random Forest and the decision tree algorithm were similar, with overall accuracy rates of 86.75% and 86.91% and the two types of F1 values were 86.12%, 86.51% and 87.04%, 86.79%.

The overall performance indexes of machine learning were lower than those of SVMs and logistic regression, of which the overall accuracy were 90.6% and 90.5% and the two F1 values were 90.76%, 90.47% and 90.54%, 90.46%, respectively. RNN and LSTM achieved better performance than traditional methods, with overall accuracy of 90.65% and 94.69%, respectively, and F1 values of 90.59%, 90.70%, 94.79% and 94.65%, respectively. The overall accuracy of the CNN_-Text_Word2-vec model is 97.6%, 7% higher than the overall accuracy of SVM. Its two F1 values are 97.73% and 97.62%, the highest two among all models. The two F1 values are 6.97% and 7.15% higher than those achieved by SVM. The overall accuracy of our model is 2.91% higher than that of the LSTM algorithm; and positive and negative class F1 values are 2.94% and 2.97% higher, respectively. Comparison of the two F1 values showsthat the CNN_Text_Word2vec model has similar effects in positive and negative classification. Experimental results show that the

Table 3
Comparison of emotional classification results .

Model	Positive *P*	Negative *P*	Positive *R*	Negative *R*	Positive F1	Negative F1	Accuracy
CNN_Text_Word2vec	95.60%	99.95%	99.95%	95.40%	97.73%	97.62%	97.60%
SVM	89.42%	91.89%	92.13%	89.1%	90.76%	90.47%	90.60%
Logistic Regression	90.18%	90.83%	90.90%	90.10%	90.54%	90.46%	90.50%
Random Forest	87.38%	85.32%	84.90%	87.73%	86.12%	86.51%	86.75%
Decision Tree	86.23%	87.63%	87.87%	85.97%	87.04%	86.79%	86.91%
Naive Bayes	79.17%	81.63%	82.37%	78.33%	80.74%	79.95%	80.35%
RNN [54]	91.12%	90.08%	90.19%	91.22%	90.59%	90.70%	90.65%
LSTM [55]	93.98%	95.53%	95.61%	93.88%	94.79%	94.65%	94.69%

Table 4
Effect of word2vec word vector on the accuracy of emotional classification .

Model	Whether the word2vec word vector is used or not	Accuracy
CNN_Text_Word2vec	No	95.41%
CNN_Text_Word2vec	Yes	97.60%
SVM	No	88.32%
SVM	Yes	90.60%
RNN	No	89.73%
RNN	Yes	90.65%
LSTM	No	93.17%
LSTM	Yes	94.69%

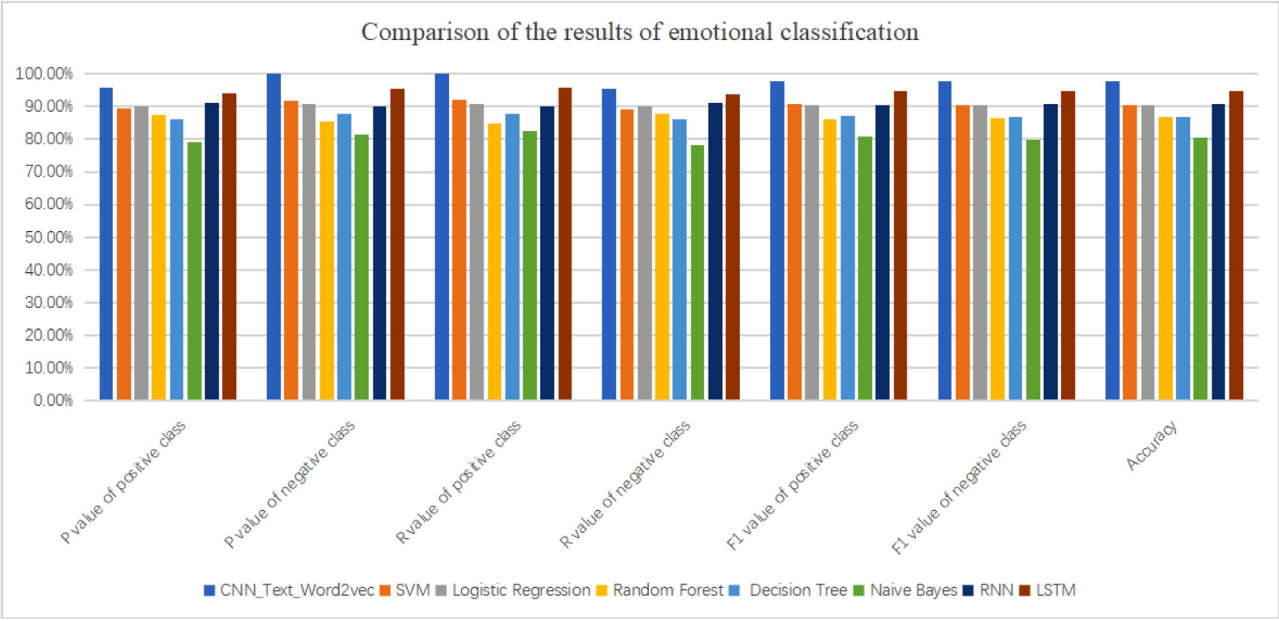


Fig. 5. Comparison of emotional classification results.

CNN_Text_Word2vec model was very effective for emotion classification of microblog short texts.

5.2.2. Experiment 2

The second set of experiments were designed to compare the overall accuracy of emotion classification whether the word2vec word vector was used or not. The results are presented in Table 4 and Fig. 6. The experimental results show that the word vectors pre-processed with word2vec in CNN_Text_Word2vec, SVM, RNN and LSTM models are more accurate than those without word2vec by 2.19%, 2.28%, 0.92% and 1.52% higher respectively. This indicates that word vectors pre-processed by word2vec can effectively improve the overall accuracy of classification, and the lower the word vector dimension is, the more effective the improvement is. By comparing CNN_Text_Word2vec, SVM, RNN and LSTM algorithms, we found that whether the word2vec word vector is used or not, the CNN_Text_-Word2vec algorithm always performed better than other algorithms, which proved effectiveness of our proposed algorithm. The experiment results showed that the pre-processed word2vec word vector can effectively improve the overall accuracy of classification, and the effect becomes obvious when the word vector dimension is low.

5.2.3. Experiment 3

The third set of experiments compared the overall accuracy of different semantic units. That is, the input of CNN was the word embedding that used characters as feature elements and the word embedding that used words as feature elements. Processing of word embedding that uses characters as feature elements was studied. Before word embedding that uses words as feature elements was processed, the word embedding was pre-processed. First, a Jieba word segmentation tool was used to seg-

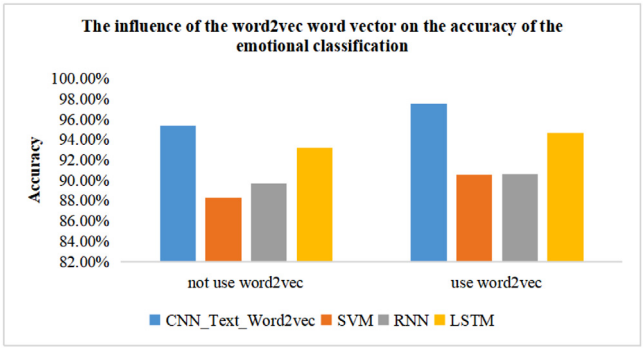


Fig. 6. Influence of word2vec word vector on the accuracy of emotional classification.

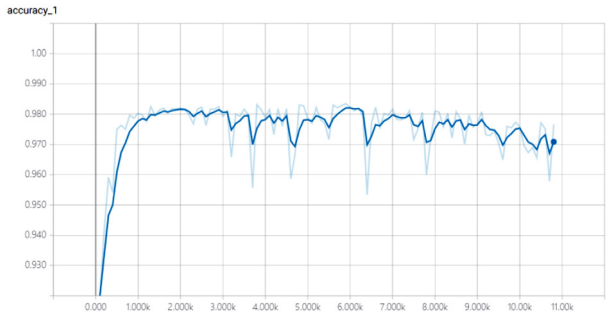


Fig. 7. Overall accuracy changes in the CNN model's word embedding of characters.

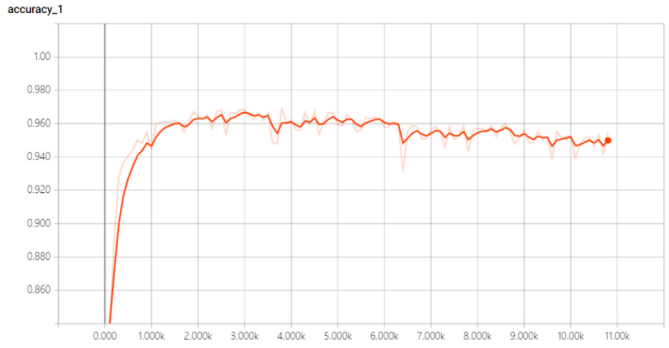


Fig. 8. Overall accuracy changes of the CNN model's word embedding of words.

ment microblog texts and separate words with spaces. Each word was trained for word embedding by the word2vec tool. A dictionary containing 24,212 items was obtained.

Verification was performed during the training in order to verify the model after every 100 training steps, and 10,800 training steps were performed. Changes in the overall accuracy of the model could be viewed on TensorBoard after the training ended. The overall accuracy of the CNN model changed under two semantic units in the verification set when the word vector dimension was 200, as illustrated in Fig. 5 and 6. The abscissa is the training step, whereas the ordinate is the accuracy.

Comparison between Fig. 7 and 8 indicates that the CNN_Text_Word2vec model basically achieved the highest accuracy in 2,000 training steps. The overall accuracy of the word embedding that used characters as feature elements is 98%, whereas the overall accuracy of the word embedding that uses words as feature elements is about 96%. In general, the CNN_Text_-Word2vec model can converge after a few training steps and achieved a high classification accuracy regardless of the word embedding used. However, the overall accuracy of the model whose word embedding uses characters as feature

Table 5
Emotional classification accuracy of the different semantic units .

Model	Semantic unit	Accuracy
CNN_Text_Word2vec	word	94.51%
CNN_Text_Word2vec	Chinese characters	97.60%
SVM	word	88.33%
SVM	Chinese characters	90.60%
RNN	word	88.57%
RNN	Chinese characters	90.65%
LSTM	word	93.54%
LSTM	Chinese characters	94.69%

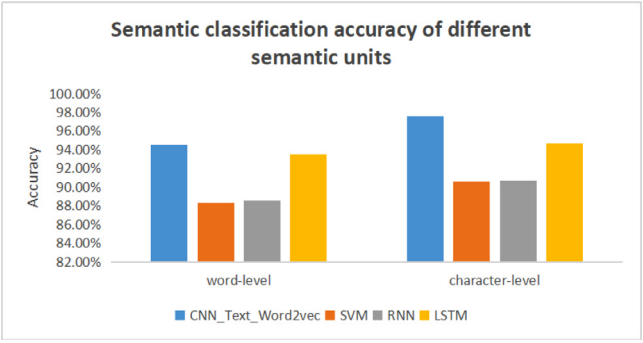


Fig. 9. Emotional classification accuracy of the different semantic units.

elements is approximately 2% higher than that achieved by the model that uses words as feature elements.

The two models were verified on the test set after training. The results are presented in Table 5 and Fig. 9. As Table 5 shows, for the CNN_Text_Word2vec, SVM, RNN and LSTM models, the overall accuracy of the word-vector model is lower than that of the character model, which was 2.9%, 2.27%, 2.12% and 1.15% lower, respectively. The model whose word embedding uses words as feature elements is low in the overall accuracy in the test set. The accuracy was 3.5%, 2.9%, and 3.1% lower than that of the model whose word embedding used characters as feature elements when the word vector dimensions were 100, 200, and 300. This result shows that the model whose word embedding used characters as feature elements is more generalized and can reduce overfitting. Thus, the feature granularity was smaller when characters were used as feature elements than when words were used as feature elements. The word embedding using characters as feature elements ccan learn more specific features than that using words as feature elements. Therefore, for the CNN_Text_Word2vec model, using Chinese characters as semantic units for microblogs did not lose semantics and can reduce over-fitting more effectively than using words as semantic units. By comparing CNN_Text_Word2vec, SVM, RNN and LSTM algorithms, we found that, for both deep learning algorithms and machine learning algorithms, the word vectors using Chinese characters as feature elements performe better than those using words. Performance of the CNN_Text_Word2vec algorithm are better than the other algorithms, thus showcasing the effectiveness of our proposed algorithm.

5.2.4. *Experiment 4*

The fourth set of experiments compared the overall accuracy of the emotion classification of models under different parameters (*Accuracy*). This group of experiments involved tuning of three parameters, namely, the gradient descent learning rate (η), the number of convolution kernels (filterNumber), and the size of the convolution kernel (filterSize). The word vector dimension of the following experiments was set to 200 after the above-mentioned experiments. Characters were used as the feature elements of the word embedding, and word2vec was used to pre-train the word embedding.

Table 6
Effects of learning rate on the accuracy of emotional classification .

Filter-Number	Filter-Size	η	Accuracy
200	3, 4, 5	0.0001	96.84%
200	3, 4, 5	0.001	97.6%
200	3, 4, 5	0.01	97.08%
200	3, 4, 5	0.05	95.2%

Table 7
Effects of convolution kernel quantity on the accuracy of emotional classification .

Filter-Number	Filter-Size	η	Accuracy
50	3, 4, 5	0.001	96.7%
100	3, 4, 5	0.001	97.3%
200	3, 4, 5	0.00	1 97.6%
300	3, 4, 5	0.001	97.4%

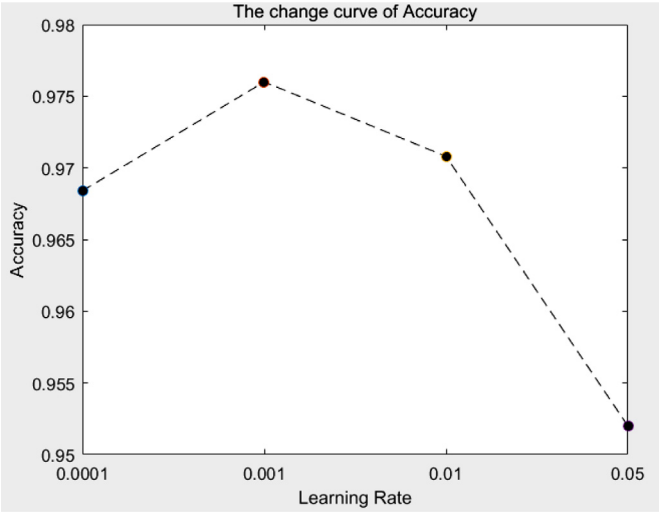


Fig. 10. Effect of learning rate on the accuracy of emotional classification.

Table 6 compares the impact of the learning rate on the accuracy of the emotion classification. In the gradient descent algorithm, if the learning rate was set too small, then the algorithm would converge slowly; if the learning rate was set too large, the cost function would oscillate. Table 6 indicates that CNN_Text_Word2vec achieved the highest overall accuracy when the learning rate was 0.001. The overall accuracy rate reduced by 0.76% when the learning rate was reduced to 0.0001. The overall accuracy rate decreased by 5.2% when the learning rate increased to 0.01. The overall accuracy dropped to 95.2% when the learning rate increased to 0.05. Fig. 10 shows the experiment results. Therefore, setting the learning rate to 0.001 would be a favorable choice. In the following experiments, the learning rate was set to 0.001.

Table 7 compares the impact of the number of convolution kernels on the accuracy of emotion classification. The results show that the number of filters (filterNumber) have no obvious influence on the accuracy. The overall accuracy rate is the highest at 97.6% when filterNumber is 200. The overall accuracy rate fluctuate around 1% with a slight change when filterNumber changed. Fig. 11 demonstrates the experiment results. Results of four sets of comparison tests confirm that CNN_Text_Word2vec has the optimal classification effect when filterNumber is 200.

Table 8 compares the impact of the convolution kernel size on the accuracy of emotion classification. The table reflects that the convolution kernel size had slight influence on the accuracy, and the accuracy fluctuated within a range of 1% as the kernel size changed, as shown

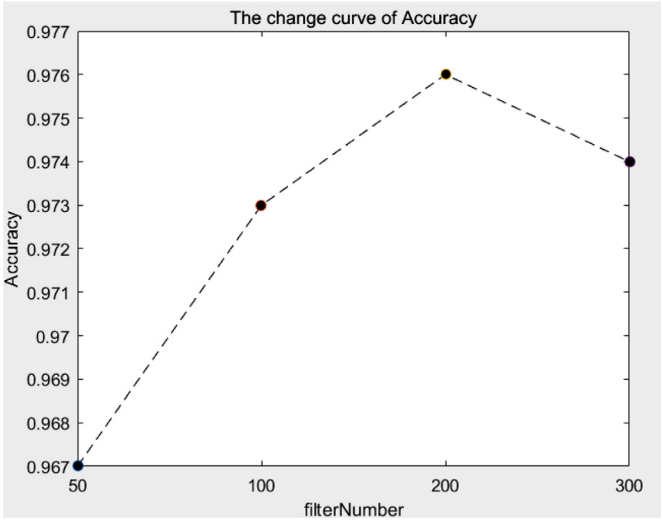


Fig. 11. Effect of convolution kernel quantity on the accuracy of emotional classification.

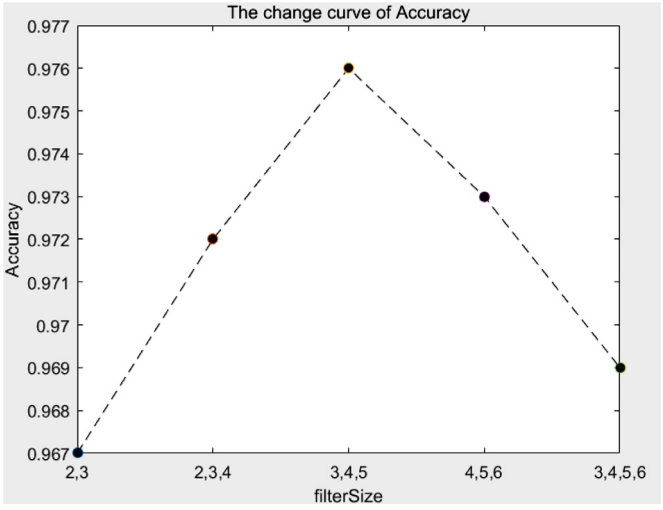


Fig. 12. Effect of convolution kernel size on the accuracy of emotional classification.

Table 8
Effects of convolution kernel quantity on the accuracy of emotional classification .

Filter-Number	Filter-Size	η	Accuracy
200	2, 3	0.001	96.7%
200	2, 3, 4	0.001	97.2%
200	3, 4, 5	0.001	97.6%
200	4, 5, 6	0.001	97.3%
200	3, 4, 5, 6	0.001	96.9%

in Fig. 12. CNN_Text_Word2vec has the optimal classification effect of 97.6% when the convolution kernel size is 3, 4, and 5 respectively.

The results of the fourth set of experiments show that the accuracy of the CNN_Text_Word2vec model fluctuated within a range of 2% as the parameter changed, and the correlation was nonlinear. The overall accuracy was 95.2%, which was 4.6% higher than that of the SVM-based emotion classification method. This result shows that CNN_Text_-Word2vec was effective in introducing the CNN into the Chinese microblog emotion classification.

6. Conclusions

This study explores the feasibility of using CNNs for the emotion analysis of microblogs. CNN_Text_Word2vec employs the word2vec neural network model to train distributed word embedding for every single word in a microblog. The trained word vectors are used as input features to learn microblog text features through parallel convolution layers with multiple convolution kernels of different sizes. The model fully considers the characteristics of short texts with sparse features and neologism of microblogs. The experiment results show that the emotional classification accuracy of CNN-_Text_word2vec on Chinese microblogs are 7%, 6.9% and 2.91% higher than that of SVM, RNN, LSTM respectively. Furthermore, in the cases of both deep learning algorithms and machine learning algorithms, the word vectors using Chinese characters as feature elements yield a higher accuracy than those using words as feature elements. Moreover, this study explores the impact of different semantic units on the models accuracy considering special features of the Chinese language. The experiment results also confirm that word embedding using characters as feature vectors performs better than using words as feature vectors.

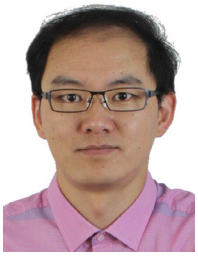
Acknowledgment

This work was supported in part by Shandong Provincial Natural Science Foundation under grant no.ZR2019PF007, Key Research and Development Plan of Shandong under grant no.2018GGX101023, the National Key Research and Development Plan of China under grant no.2018YFB0803504, Basic Scientific Research Operating Expenses of Shandong University under grant no.2018ZQXM004, Guangdong Province Key Research and Development Plan under grant no. 2019B010137004 and Guangdong Province Universities and Colleges Pearl River Scholar Funded Scheme (2019).

References

[1] B. Pang, L. Lee, Opinion mining and sentiment analysis, Found. Trends Inf. Retrieval 2 (12) (2008) 1–135.
[2] H. Peng, E. Cambria, A. Hussain, A review of sentiment analysis research in Chinese language, Cognit. Comput. 9 (4) (2017) 423–435.
[3] L. Gui, Y. Zhou, R. Xu, et al., Learning representations from heterogeneous network for sentiment classification of product reviews, Knowl. Based Syst. 124 (2017) 34–45.
[4] E. Cambria, Affective computing and sentiment analysis, IEEE Intell. Syst. 31 (2) (2016) 102–107.
[5] S. Rosenthal, N. Farra, P. Nakov, Semeval-2017 task 4: sentiment analysis in twitter, in: Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017), 2017, pp. 502–518.
[6] E. Breck, C. Cardie, Opinion Mining and Sentiment Analysis, The Oxford Handbook of Computational Linguistics 2nd edition, 2017.
[7] Y. Rao, J. Lei, L. Wenyin, et al., Building emotional dictionary for sentiment analysis of online news, World Wide Web 17 (4) (2014) 723–742.
[8] E. Cambria, S. Poria, D. Hazarika, et al., Senticnet 5: discovering conceptual primitives for sentiment analysis by meansof context embeddings, in: Thirty-Second AAAI Conference on Artificial Intelligence, AAAI, 2018.
[9] G. Haralabopoulos, E. Simperl, Crowdsourcing for beyond polarity sentiment analysis a pure emotion lexicon, 2017, 2017. arXiv:1710.04203.
[10] S. Buechel, S. Rücker, U. Hahn, Learning and evaluating emotion lexicons for 91 languages, To Appear in ACL, 2020.
[11] X. Cheng, Y. Chen, B. Cheng, S. Li, G. Zhou, An emotion cause corpus for chinese microblogs with multiple-user structures, ACM Trans. Asian Low-Resource Lang. Inf. Process. 17 (1) (2017) 1–19.
[12] Y. Chen, W. Hou, X. Cheng, S. Li, Joint learning for emotion classification and emotion cause detection, in: Proc. 2018 Conf. Empir. Methods Nat. Lang. Process. EMNLP, 2018, pp. 646–651. 2020
[13] Q. Dang, F. Gao, Y. Zhou, Early detection method for emerging topics based on dynamic Bayesian networks in micro-blogging networks, Expert Syst. Appl. 57 (2016) 285–295.
[14] H. Krishnan, M.S. Elayidom, T. Santhanakrishnan, Emotion detection of tweets using naive bayes classifier, Emotion (2017).
[15] B. Pang, L. Lee, S. Vaithyanathan, Thumbs up: sentiment classification using machine learning techniques, in: Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing-Volume 10. Association for Computational Linguistics, 2002, pp. 79–86.
[16] T. Mullen, N. Collier, Sentiment analysis using support vector machines with diverse information sources, in: Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, 2004.
[17] A. Kennedy, D. Inkpen, Sentiment classification of movie reviews using contextual valence shifters, Comput. Intell. 22 (2) (2006) 110–125.
[18] A. Abbasi, H. Chen, A. Salem, Sentiment analysis in multiple languages: feature selection for opinion classification in web forums, ACM Trans. Inf. Syst. (TOIS) 26 (3) (2008) 12.
[19] P.D. Turney, Thumbs up or thumbs down: semantic orientation applied to unsupervised classification of reviews, in: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics. Association for Computational Linguistics, 2002, pp. 417–424.
[20] T. Wilson, J. Wiebe, P. Hoffmann, Recognizing contextual polarity in phrase-level sentiment analysis, in: Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2005, pp. 347–354.
[21] A. Adreevskaia, S. Bergler, Mining wordnet for a fuzzy sentiment: sentiment tag extraction from wordnet glosses, in: The 11th Conference of the European Chapter of the Association for Computational Linguistics, 2006.
[22] Y. Lu, M. Castellanos, U. Dayal, et al., Automatic construction of a context-aware sentiment lexicon: an optimization approach, in: Proceedings of the 20th International Conference on World Wide Web. ACM, 2011, pp. 347–356.
[23] T. Zagibalov, J. Carroll, Automatic seed word selection for unsupervised sentiment classification of Chinese text, in: Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1. Association for Computational Linguistics, 2008, pp. 1073–1080.
[24] V. Sindhwani, P. Melville, Document-word co-regularization for semi-supervised sentiment analysis, in: The 8th IEEE International Conference on Data Mining (ICDM'08). IEEE, 2008.
[25] S. Dasgupta, N. Vincent, Mine the easy, classify the hard: a semi-supervised approach to automatic sentiment classification, in: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2. Association for Computational Linguistics, 2009.
[26] X. Wan, Co-training for cross-lingual sentiment classification, in: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1. Association for Computational Linguistics, 2009.
[27] S. Li, et al., Semi-supervised learning for imbalanced sentiment classification, in: IJCAI Proceedings-International Joint Conference on Artificial Intelligence, 22, 2011.
[28] J. Janai, F. Güney, A. Behl, et al., Computer vision for autonomous vehicles: problems, datasets and state-of-the-art, 2017, arXiv:1704.05519.
[29] K.P. Braho, J.P. Pike, L.A. Pike, Methods and systems for identifying errors in a speech recognition system, U. S. Patent 9 (928) (2018) 3–27. 829
[30] Y. Bengio, et al., A neural probabilistic language model, Journal of machine learning research (2003). 3.feb
[31] T. Mikolov, et al., Efficient estimation of word representations in vector space, 2013a, arXiv:1301.3781.
[32] T. Mikolov, et al., Distributed representations of words and phrases and their compositionality, Adv. Neural Inf. Process. Syst. (2013).
[33] X. Glorot, A. Bordes, Y. Bengio, Domain adaptation for large-scale sentiment classification: a deep learning approach, in: Proceedings of the 28th International Conference on Machine Learning (ICML-11), 2011.
[34] R. Socher, et al., Recursive deep models for semantic compositionality over a sentiment treebank, in: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, 2013.
[35] D. Tang, F. Wei, N. Yang, et al., Learning sentiment-specific word embedding for twitter sentiment classification, in: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2014, pp. 1555–1565.
[36] Y. Ren, Y. Zhang, M. Zhang, et al., Context-sensitive twitter sentiment classification using neural network, in: Thirtieth AAAI Conference on Artificial Intelligence, 2016.
[37] P. Kim, Convolutional neural network [M]. MATLAB Deep Learning, Apress, Berkeley, CA, 2017. 121-147
[38] Z. Tian, W. Shi, Y. Wang, C. Zhu, X. Du, S. Su, Y. Sun, N. Guizani, Real time lateral movement detection based on evidence reasoning network for edge computing environment, IEEE Trans. Ind. Inf. (2019), doi:10.1109/TII.2019.2907754.
[39] Z. Tian, M. Li, M. Qiu, Y. Sun, S. Su, Block-DEF: a secure digital evidence framework using blockchain, Inf. Sci. (Ny) 491 (2019) 151–165, doi:10.1016/j.ins.2019.04.011.
[40] J. Wehrmann, W. Becker, H.E.L. Cagnini, et al., A character-based convolutional neural network for language-agnostic twitter sentiment analysis, in: Neural Networks (IJCNN), 2017 International Joint Conference on. IEEE, 2017, pp. 2384–2391.
[41] M. Volpi, D. Tuia, Dense semantic labeling of subdecimeter resolution images with convolutional neural networks, IEEE Transactions on Geoscience and Remote Sensing (55) (2017) 881–893. 2
[42] S. Poria, H. Peng, A. Hussain, et al., Ensemble application of convolutional neural networks and multiple kernel learning for multimodal sentiment analysis, Neurocomputing 261 (2017) 217–230.
[43] Y. Goldberg, Neural network methods for natural language processing, Synthesis Lect. Hum. Lang. Technol. 10 (1) (2017) 1–309.
[44] Z. Tian, Y. Cui, L. An, S. Su, X. Yin, L. Yin, X. Cui, A real-time correlation of host-level events in cyber range service for smart campus, IEEE Access 6 (2018) 35355–35364, doi:10.1109/ACCESS.2018.2846590.
[45] R. Satapathy, .Y. Li, S. Cavallari, E. Cambria, Seq2seq deep learning models for microtext normalization, in: In 2019 International Joint Conference on Neural Networks (IJCNN), Pp. 1–8. IEEE, 2019.

[46] X. Wang, W. Jiang, Z. Luo, Combination of convolutional and recurrent neural network for sentiment analysis of short texts, in: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, 2016, pp. 2428–2437.
[47] M. Arora, V. Kansal, Character level embedding with deep convolutional neural network for text normalization of unstructured data for twitter sentiment analysis, Soc. Netw. Anal. Min. 9 (1) (2019) 12.
[48] Y. Kim, Convolutional neural networks for sentence classification, 2014, arXiv:1408.5882.
[49] T. Mikolov, W.-t. Yih, G. Zweig, Linguistic regularities in continuous space word representations, in: Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2013.
[50] Z. Tian, X. Gao, S. Su, J. Qiu, X. Du, M. Guizani, Evaluating reputation management schemes of internet of vehicles based on evolutionary game theory, IEEE Trans. Veh. Technol. 10.1109/TVT.2019.2910217.
[51] Z. Tian, S. Su, W. Shi, X. Du, M. Guizani, X. Yu, A data-driven method for future internet route decision modeling, Future Gen. Comput. Syst. 95 (2019) 212–220.
[52] R. Collobert, et al., Natural language processing (almost) from scratch, Journal of Machine Learning Research (2011) 2493–2537. 12.Aug
[53] Q. Tan, Y. Gao, J. Shi, X. Wang, B. Fang, Z. Tian, Towards a comprehensive insight into the eclipse attacks of tor hidden services, IEEE Internet Things J. (2018), doi:10.1109/JIOT.2018.2846624.
[54] R. Socher, B. Huval, C.D. Manning, A.Y. Ng, Semantic compositionality through recursive matrix-vector spaces, in: In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-coNLL '12), 2012, pp. 1201–1211.
[55] K.S. Tai, R. Socher, C.D. Manning, Improved semantic representations from tree-structured long short-term memory networks, (2015). arXiv:1503.00075.
[56] G.E. Hinton, et al., Improving neural networks by preventing co-adaptation of feature detectors, 2012, arXiv:1207.0580.



Dongliang Xu, lecturer, ShanDong University (WeiHai). He received Ph.D. degrees from Harbin Institute of Technology, Harbin, China, in 2015. His research interests include Network Security,Artificial Intelligence Security,Big Data Security.

Zhihong Tian, Ph.D., professor, PHD supervisor, Dean of cyberspace institute of advanced technology, Guangzhou University. Standing director of CyberSecurity Association of China. Member of China Computer Federation. From 2003 to 2016, he worked at Harbin Institute of Technology. His current research interest is computer network and network security.

Rufeng Lai, he received his B.E degree from Shandong University. And now he is a M.Sc. candidate of Beihang University, Beijing, China. His research interests include data mining, information security.

Xiangtao Kong, undergraduate student, Shandong University (Weihai). His research interests are machine learning and computer vision.

Dr Zhiyuan (Thomas) Tan is a Lecturer at the School of Computing at the Edinburgh Napier University (ENU), the United Kingdom. His current research interests include cybersecurity, machine learning, data analytics, virtualisation, and cyber-physical system. Dr Tan is now working on the following research topics: (1) Adversarial machine learning for Anomaly/Malware detection; (2) Virtualisation security based on non-parametric behaviour modelling; (3) Knowledge transfer (Transfer Machine Learning) in cyber-security problems; and (4) IoT Security with focuses on Cloud and Edge computing security issues.

Dr. Wei Shi is an Associate professor in the School of Information Technology, cross appointed to the department of Systems and Computer Engineering, in the Faculty of Engineering and Design, at Carleton University in Ottawa, Canada. She is specialized in the design and analysis of fault tolerance algorithms addressing security issues in distributed environments such as Data-Center Networks, Clouds, Mobile Agents and Actuator Systems, Wireless Sensor Networks, as well as Critical Infrastructures such as Power Grids and Smart Cities. She has also been conducting research in data privacy and Big Data analytics. Wei holds a Bachelor of Computer Engineering from Harbin Institute of Technology (HIT), as well as a Master's and a Ph.D. of Computer Science from Carleton University. She is also a Professional Engineer licensed in Canada.