Classification for Social Media Short Text Based on Word Distributed Representation

Dexin Zhao, Zhi Chang(✉), Nana Du, and Shutao Guo

Tianjin University of Technology, Tianjin 300384, China
qiqiharxin@163.com, changzhi1123@outlook.com, cloud-dn@163.com, alsonsmileshine@hotmail.com

Abstract. According to the brief and meaningless features of the social media content, we propose a text classification algorithm based on word vectors, which can quickly and effectively realize the automatic classification of the short text in social media. In view of the lack of word order and position considerations in the Word2vec model, we combine the Word2vec trained word vector with the convolutional neural network (CNN) model to propose SW-CNN and WW-CNN classification algorithms. The methods are evaluated on the three different datasets. Compared with existing text classification methods based on convolutional neural network (CNN) or recurrent neural network (RNN), the experimental results show that our approach has superior performance in short text classification.

Keywords: Text classification · Word2vec · Word vector
Social media

1 Introduction

With the rapid development of mobile terminals, a large number of social software has become an indispensable part of people's lives. Although people's lives become more convenient, the social media data generated by the users have made it difficult to extract useful information to fully satisfy the needs of people. Most of the social media produces short texts that look like spoken texts and more casual than traditional text. Social media short text data stream contains a lot of noise and unstructured information, slang and lack of words. These have brought great difficulty to the classification of the short text in social media.

In the traditional text classification problems, there have been a lot of related research, such as classification method based on rule characteristics [1,2], a classification method combined with SVMs and Naive Bayesian [3], classification method of building dependency tree based on the conditional random field [4]. In recent years, deep learning has shown good results in image processing and Natural Language Processing tasks, including convolutional neural network (CNN)

X. Meng et al. (Eds.): WISA 2018, LNCS 11242, pp. 259–266, 2018.
https://doi.org/10.1007/978-3-030-02934-0_24

[5,6] and recurrent neural network (RNN) [7]. In general, CNN is used to deal with the problem of image classification, convolution and pooling structure can extract various textures and features in images and finally integrate and output information with the fully connected network. In short text analysis tasks, due to the compact structure and independent expression of meaning, CNN becomes possible in dealing with this kind of problem. The author Yoon Kim [5] based on distributed word vector to represent text by using a simple CNN network to classify text from two aspects of specific tasks and static vectors and achieved excellent results. Rie Johnson and Tong Zhang [8] take into account the problem of word order, they improved the word bag model and classified the text by the convolution layer of CNN and proposed the seq-CNN and bow-CNN models. [11] propose a new neural language model incorporating both word order and character order in its embedding. Chunting Zhou *et al.* [9] and Ji Young Lee *et al.* [10] studied a combination of CNN and RNN respectively. The former uses the combination of CNN and RNN as text classifier; the latter uses CNN and RNN to train the vectors and uses the common artificial neural network (ANN) as a classifier. These classification methods using deep learning have also achieved excellent results.

However, since social media short text has the characteristics of short length, huge interference, irregular, features sparseness and other characteristics, these traditional text feature representation methods based on vector space model are difficult to obtain satisfying text representation results. Although word vector representation technology based on deep learning can well express the grammatical and semantic relations of words, it loses the word order information in training process and does not take into account the influence of word order on text meaning.

Based on the above problems, this paper begins with words feature representation, studies on social media short text classification based on Word2vec model and Convolution Neural Network model (CNN). Given the lack of word order and position considerations in the Word2vec and the CNN model, we propose w-Word2vec(WW) and seq-Word2vec(SW) algorithm. Finally, it has input the word vector that related to the word order and the position to the CNN model for training.

2 Model

2.1 Word Vector Model Based on Word Order

In previous research, Rie Johnson and Tong Zhang [8] have improved the one-hot expression according to the word order, but because the one-hot expression does not consider the semantic information between words like word2vec, it's too redundant and sparse. As a result, we use their idea to improve the word2vec model based on the problem of word order information.

We add the word vectors of the two words that are connected before and after about word order problem, as shown in Fig. 1, the sentence "Please do not tell me that", we have word vectors of the adjacent words"please" and "do","do"

and "not", "not" and "tell", and so on, adding two word vectors together, and then implementing a consideration of word order by this way. We assume that a sentence is composed of, and the formula is as follows:

$$x_w = e(w_i) + e(w_{i+1}), \begin{cases} w_i \in c \\ i \neq k \end{cases} \tag{1}$$

Among them, the c represents all the context words of the word w, and e(w_i) represents the word vector of the word w_i. This method is called seq-word2vec.

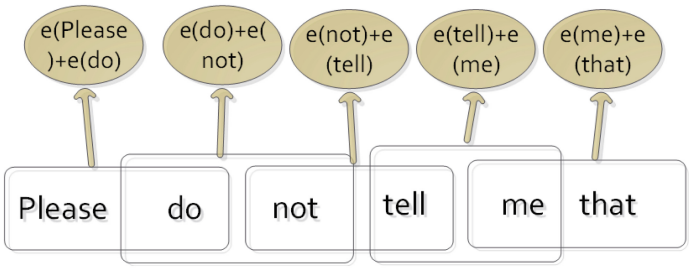

Fig. 1. Seq-word2vec model.

In addition to the above improvement, we also consider another situation: in two adjacent words, the word of the front is the vector of its own word, and the latter is not the first word because of its position. According to the previous example, the sentence "Please do not tell me that", the original word vector of the word"please" plus the original word vector of the word "do" that represents the word vector of the word"please", the original word vector of the word "do" plus the original word vector of the word"not" that represents the word vector of the word "do". Now we consider the position factor, adding weight to the word vector, the update formula is as follows:

$$x_w = e(w_i) + ae(w_{i+1}), \begin{cases} w_i \in c \\ i \neq k \end{cases} \tag{2}$$

Among them, a is the weight of the second word in the two adjacent words, it represents the important relationship between two adjacent words. The method structure shown in the following figure, which defines the method named w-word2vec.

2.2 CNN for Text Classification

Now we consider the application of CNN in text classification. CNN can be widely applied to text classification, CNN model is similar to the N-gram model, and filter window in CNN can be seen as a N-gram method. Besides, the use of

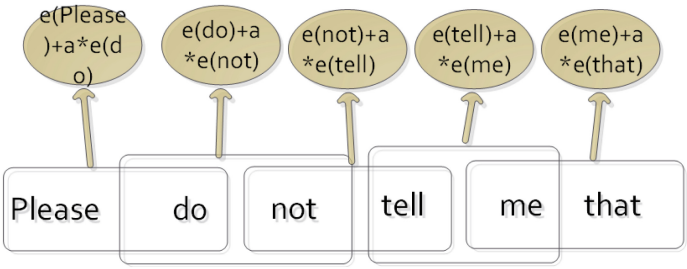

Fig. 2. w-word2vec model.

convolution layer and the pooling layer in CNN reduces the number of training parameters and extracts information at the higher level (Fig. 2).

In 2014, the application of CNN proposed by Kim *et al.* [5]. For text classification tasks. In recent years, CNN has been used as a baseline method, we will make improvements on this basis and classify short texts of social media.

If a document $D = (s_1, s_2, \ldots, s_k)$, s represents the sentence in the document, the number of different words is N, and the word vector dimension of every word is d. Since each sentence has a different length, therefore, we choose a unified sequence of words with a sentence length K, document D has N words without repetition, the length of each sentence is K, and the word vector dimension of each word is d, we set the word vector of the i-th word as e_i. We use the word vector of each sentence as the input of the CNN model, that is, the matrix of the $k \times d$ dimension.

2.3 Text Classification Model Based on Word Vector

Previous work shows that the word2vec model and the CNN model has a good effect on Natural Language Processing, but the word2vec model does not take into account the word order. Therefore, for the characteristics of social media short text own short length, huge interference, irregular, features sparseness, we combine CNN model and improved word vector to classify social media short texts. In next section, we will test the methods proposed in this paper. This section mainly introduces the model of the combination of word vectors considering word order with CNN. The model structure is shown in Fig. 3.

First, considering the input text data in CNN model are the word vector of each sentence, a sentence is expressed as matrix of dimension $k \times d$, K is the number of words in this sentence, D is the word vector dimension of every word, then the matrix input to CNN. We change the word vector to the two word vectors considered in word order in the previous chapter, respectively.

We use the improved word vector matrix as the input matrix of CNN, then we can see the convolution layer consisting of multiple filters. Here, we refer to the setting of the filter by Kim *et al.* [5], the size of the three filters are 3, 4, 5 respectively. They represent 3 words, 4 words, and 5 words, respectively, and

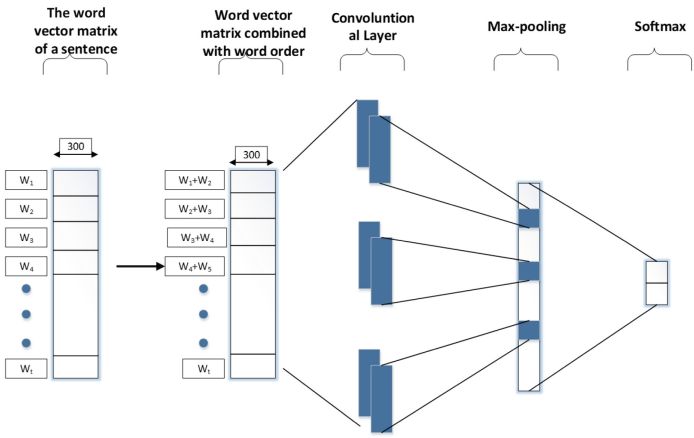

Fig. 3. SWCNN model.

according to the size of different filters, convolution operations are carried out with a variety of filters, and the corresponding local features of different sizes are obtained. In Natural Language Processing, some crucial keywords can reflect the category of this sentence. In this paper, we also choose the best feature of each feature dictionary, the features have been extracted into the pool layer, through the use of the pool layer, the model further selects the most representative features of each feature information of the dictionary. We select the max-pooling, which is to record only the maximum value of each feature dictionary. The pool layer can reduce not only the number of model parameters but also filter some noise. In this case, generating a string of single variable feature vector from each dictionary, and then splicing these characteristics obtained by pooling and form a long feature vector. Then, the feature vector enters the full connection layer and add dropout regularization during the training process of the fully connected layer to prevent overfitting. Finally, using the regularized feature vector as input and sentences classification by softmax classification function, where $softmax(z_i) = e^{z_i} / \sum_j e^{z_j}$. Finally, categorizing the sentence to the category with the highest probability to finish the text classification.

2.4 Loss Function

Using the model outputs we can define the loss function. The loss is a measurement of the error our network makes, and our goal is to minimize it. The standard loss function for categorization problems it the cross-entropy loss.

$$L(p, q) = -\sum_x p(x) \log q(x) \tag{3}$$

where $p(x)$ represents the true probability distribution and $q(x)$ represents the probability distribution of the prediction.

3 Evaluation and Analysis

3.1 Dataset and Evaluation Metric

Binary and Multi-Label Classification are used to prove the test result in this paper. Multi-Label Classification dataset: Twitter data is used as data sources in this test, a total of 10539 pieces in three topics, among which 90% of each classification was used for training while the remaining 10% Twitter data was used for testing. The three topics are disaster, auto, and politics.

Binary Classification dataset: Movie Review data from Rotten Tomatoes are used for binary classification. It includes 10662 reviews, among which 5331 reviews are positive and 5331 reviews are negative.

Accuracy is used to evaluate effect of the classifier. Accuracy is for evaluation of all data. No matter which class it is, as long as it is correct, it is molecule with total amount of the data being the denominator. Or we can say, accuracy is an evaluation of the correct rate on the classifier as a whole.

$$Accuracy = (TP + TN)/(P + N) \tag{4}$$

TP is the number of cases being correctly classified as the positive case, TN is the number of cases being correctly classified as the negative case; P and N represent the amount of all positive and negative cases.

3.2 Experimental Design

(1) Filter all punctuation marks and special characters.
(2) Managed all non-standard text in original source, such as messy code or English character of full width. The non-standard text in the original source should be converted. There is no need to separate word because the text we chose is in English which uses space to separate word.
(3) Word embedding training was carried out with word2vec model in gensim in python to input continuous sense for training to generate word embedding with the size of 300.
(4) Expand the processed sentences in Step 2 to the longest to construct glossary index and then reflect each word to an integer with the size of the glossary to make each sentence an integer embedding.
(5) Correspond integer of each sentence to the trained word embedding in Step 3 to make a word embedding matrix and then process word embedding according to seq-word2vec and w-word2vec to generate new word embedding matrix.
(6) Finally, input the word embedding matrix generated in Step5 into CNN for training to get the final classification.

3.3 Experimental Results and Analysis of the Result

The comparison methods used in this paper are all deep learning related methods. The experiment test six different classification methods, including the seq-word2vec+CNN and w-word2vec+CNN methods proposed in this paper, which

we referred to as SW-CNN and WW-CNN, respectively. Moreover, the performance of these six methods results are shown below:

Table 1. Accuracy.

Model	Binary classification	Multi classification
Char-CNN	72.7%	94.7%
FastText	79.9%	96.2%
LSTM	75.3%	97.3%
CNN	72.8%	95.4%
SW-CNN	75.2%	98.1%
WW-CNN	75%	98.7%

The second column in Table 1 is the accuracy of movie review classification with the six classification approaching. We can see that the accuracy for Char-CNN is 72.7%, the lowest among the six, followed by 72.8% of CNN. The best is the FastText with accuracy of 79.9% while that of SW-CNN proposed in this paper is 75.2%. Accuracy of WW-CNN, 75.0%, is slightly lower than that of SW-CNN. The thrid column in Table 1 is the comparison of the six classification approaches in three classes of Twitter data sources, from which we can see that the two approaches proposed in this paper enjoy the highest accuracy among the rest, 2.7% and 3.3% higher than that of CNN text, followed by LSTM, with FastText being the worst. Accuracies of the six approaches are all higher than 90%.

Compared with traditional classification approaches, our two approaches proposed in this paper had good improvement on classification of short text. Both improved word embedding via word order relationship. Moreover, the WW-CNN model takes into account the importance of location, and then tests on two datasets show that the method proposed in this paper is better than the previous CNN method. In terms of Twitter data sources, their performance compared with the rest four approaches is exhibited, with the accuracy of WW-CNN being slightly higher than that of the SW-CNN. For other classification method, we need to make specific judgments based on different datasets.

The reason for the above is summarized as follows: First, embeddings learned from the very beginning were not used, which was replaced by that trained in Word2vec. The processed embeddings are of higher group characters than those random ones. Second, the original word embedding didn't consider the relationship between words while the text considered it and the major factors of different position are considered in WW-CNN. Finally, text processed by CNN can express text characteristics better.

4 Conclusion

In this paper, we use a word vector considering word order relation as input, and then use CNN to classify text. Different from the traditional word vector model,

although the word2vec model contains the semantic features between words, it can better express the text features, but it does not take into account the relationship between word order. We propose SW-CNN and WW-CNN model to improve the expression of word vectors. Experiments show that the proposed method (SW-CNN) algorithm and WW-CNN algorithm) have increased the accuracy of 2.7% and 3.3% compared with that of traditional CNN algorithm on Multi-label classification of short texts in social media.

References

1. Silva, J., Coheur, L., Mendes, A.C.: From symbolic to sub-symbolic information in question classification. Artif. Intell. Rev. **35**(2), 137–154 (2011)
2. Huang, Z., Thint, M., Qin, Z.: Question classification using head words and their hypernyms. In: EMNLP, pp. 927–936 (2008)
3. Wang, S., Manning, C.D.: Baselines and bigrams: simple, good sentiment and topic classification. In: Association for Computational Linguistics, pp. 90–94 (2012)
4. Delaye, A., Liu, C.L.: Text/Non-text classification in online handwritten documents with conditional random fields. Commun. Comput. Inf. Sci. **321**(53), 514–521 (2012)
5. Kim, Y.: Convolutional neural networks for sentence classification. In: Conference on Empirical Methods on Natural Language Processing (2014)
6. Kalchbrenner, N., Grefenstette, E., Blunsom, P.: A Convolutional neural network for modelling sentences. Eprint Arxiv. (2014)
7. Socher, R., Huval, B., Manning, C.D., et al.: Semantic compositionality through recursive matrix-vector spaces. In: Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 1201–1211 (2012)
8. Johnson, R., Zhang, T.: Effective use of word order for text categorization with convolutional neural networks. Eprint Arxiv (2014)
9. Zhou, C., Sun, C., Liu, Z., et al.: A C-LSTM neural network for text classification. Comput. Sci. **1**(4), 39–44 (2015)
10. Lee, J.Y., Dernoncourt, F.: Sequential short-text classification with recurrent and convolutional neural networks. 515–520 (2016)
11. Trask, A., Gilmore, D., Russell, M.: Modeling order in neural word embeddings at scale. Comput. Sci., 2266–2275 (2015)