Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 9340194, 6 pages
https://doi.org/10.1155/2018/9340194

Research Article

Improved Convolutional Neural Network for Chinese Sentiment Analysis in Fog Computing

Haoping Chen, Lukun Du, Yueming Lu, and Hui Gao

Key Laboratory of Trustworthy Distributed Computing and Service (BUPT), Ministry of Education, Beijing University of Posts and Telecommunications, Beijing, China

Correspondence should be addressed to Haoping Chen; chenhaopingbupt@163.com

Received 22 June 2018; Accepted 15 August 2018; Published 23 September 2018

Academic Editor: Fuhong Lin

Fog computing extends the concept of cloud computing to the edge of network to relieve performance bottleneck and minimize data analytics latency at the central server of a cloud. It uses edge nodes directly to perform data input and data analysis. In public opinion analysis system, edge nodes that collect opinions from users are responsible for some data filtering jobs including sentiment analysis. Therefore, it is crucial to find suitable algorithm that is lightweight in operation and accurate in predictive performance. In this paper, we focus on Chinese sentiment analysis job in fog computing environment and propose a non-task-specific method called Channel Transformation Based Convolutional Neural Network (CTBCNN) for Chinese sentiment classification, which uses a new structure called channel transformation based (CTB) convolutional layer to enhance the ability of automatic feature extraction and applies global average pooling layer to prevent overfitting. Through experiments and analysis, we show that our method do achieve competitive accuracy and it is convenient to apply this method to different cases in operation.

1. Introduction

The concept of fog computing [1, 2] is extended from cloud computing which puts a substantial amount of data analysis to edge nodes. These edge nodes are densely geographically deployed and are close to original data input. The main purpose of fog computing is to relieve network traffic load and reduce data calculation latency at the cloud. With the explosive growth of data from the Internet, fog computing is getting more and more important in Internet of Things (IoT) [3], Intrusion Detection, and many other fields. In a conceptual framework of public opinion analysis system, edge nodes are responsible for some data filtering jobs including sentiment analysis. Therefore, operations on these edge nodes are supposed to be lightweight and accurate in predictive performance.

Sentiment analysis is an important task in many real-world applications and there are many researches on it. Most researches on sentiment analysis mainly divide into two categories: unsupervised methods based on sentimental lexicon and supervised machine learning methods. The first strategy identifies polarity of text using sentiment lexicons. Reference [4] implemented sentiment value calculation and classification of microblog texts by extensible sentiment dictionary. Reference [5] combined basic emotion value lexicon and social evidence lexicon to improve traditional polarity lexicon, thus achieving significant improvement in Chinese text sentiment analysis. There are various ways to express the same opinion in Chinese. Therefore, it is impossible for any lexicon to cover all sentiment words or phrases. Furthermore, the same word in different field may have different sentimental impacts, which means lexicon is usually task-specific, while supervised machine learning methods use traditional text classification methods, such as Naive Bayes (NB) and Support Vector Machine (SVM). Pang applied several machine learning techniques to the sentiment classification problem in [6], which used movie reviews as data and the results showed that SVM outperform other methods. Feature extraction methods based on TF-IDF, Mutual Information, and Chi-Square [7] are commonly used for machine learning methods. However, classification performance varies a great deal with different feature selection methods [8]. Meanwhile, feature engineering is also task-specific. As far as we know, most sentiment analysis methods are implemented at cloud

server. It is crucial to find a suitable algorithm that works in fog computing environment. Recently, more and more researchers apply deep neural networks to sentiment analysis [9–11]. And the release of TensorFlow Lite makes it possible to run deep learning models on portable equipment.

In this work, we focus on Chinese sentiment analysis and aim at presenting a lightweight, non-task-specific method with high portability in fog computing environment without manual feature engineering. We propose a method called Channel Transformation Based Convolutional Neural Network (CTBCNN), which is an improved model of Convolutional Neural Network (CNN). Firstly, we input sentences and obtain word vectors by skip-gram model [12, 13] and keep them static. Then we utilize a new structure of called channel transformation based (CTB) convolutional layer, which enhances the capability of automatic feature extraction by considering the channel information of the output of the previous convolutional layer. Our method replaces the fully connected layer by global average pooling layer to prevent overfitting. Global average pooling was proved effective in computer vision tasks by Lin Min [14]. Inspired by Lin M's work, we study the outputs by global average pooling layer instead of fully connected layer.

The main contributions of this work are presented as follows. Firstly, we present an effective method that is suitable for fog computing environment. The method can be applied to different cases to handle different data conveniently without much human operations. Secondly, we put forward the idea of channel transformation for convolution layer, which is able to cover more information and extract more representative feature. Thirdly, our work proves the regularization effect of global average pooling layer in nature language processing task.

This work is organized as follows. Section 2 introduces the overall model of CTBCNN. Section 3 shows the structure of the stacking of CTB convolutional layers. Section 4 presents the implementation of global average pooling layer. Section 5 shows the experimental results. We conclude this work in Section 6.

2. Channel Transformation Based Convolutional Neural Network

We first present the overall model structure of CTBCNN model, which is presented in Figure 1. Compared with classic CNN [15], CTBCNN has two main differences. Firstly, CTBCNN has three CTB convolutional layers. Channel transformation is a trick of matrix transformation that allows us to implement convolution not only on height×width plane but also on height×depth plane and depth×width plane. This kind of convolution enhances capability of feature extraction by covering more information in channel depth dimension. The second difference is that we replace the fully connected layer with global average pooling layer because fully connected layer is a kind of dense connection which is prone to overfitting.

CTBCNN takes sentence vector as input and extracts feature maps from each input sentence by the three CTB convolutional layers and then utilize global average pooling over these feature maps to output the sentiment classification result.

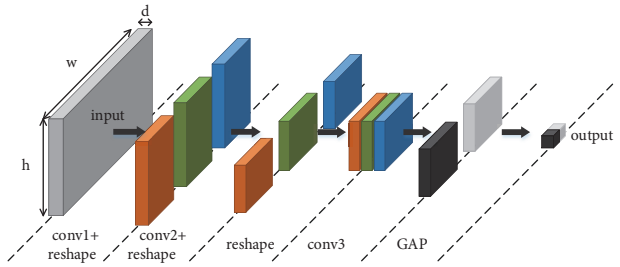


Figure 1: The overall structure of Channel Transformation Based Convolutional Neural Network. The model basically consists of three channel transformation based convolutional layers and a global average pooling (GAP) layer. This paper focuses on binary classification and the number of output categories is 2.

In the following two sections, we will present the structure of the three channel transformation based convolutional layers and global average pooling layer in detail.

3. Channel Transformation Based Convolutional Layer

In this section, we first give the definition of the shape of a matrix or a vector. In the second part we introduce the conventional convolutional layer and convolutional process. The third part presents the CTB convolutional layer on the basis of conventional convolutional layer. Finally we stack CTB convolutional layers and give the three CTB convolutional layers structure.

3.1. Definition of Shape. To make it more understandable, we define the shape of a matrix or vector using three dimensions: height, width, and depth. The input and output result can be represented by their shape in an intuitive manner of the convolutional layer and, for example, we have an image which size is h×w and has RGB 3 channels; then the shape of the image can be represented as (h, w, 3).

3.2. Conventional Convolutional Layer. The input is a sentence vector that can be represented as

$$x_{1:n} = x_1 \oplus x_2 \oplus \ldots \oplus x_n \tag{1}$$

where $x_i \epsilon R^k$ is a k dimensional word vector of the i^{th} word in sentence that consists of n words and $\oplus$ means concatenation operation. Thus the input sentence vector is similar to an "image" whose shape is (n, k, 1).

Generally in convolution process, a filter $W \epsilon R^{h_c \times k}$ like a window of h_c words is applied to generate a new feature c_i.

$$c_i = f\left(W \bullet x_{i:i+h_c-1} + b\right) \tag{2}$$

where $b \epsilon R$ is a bias term and f is an activation function. In this work we use ReLU as the activation function. From [16] we know that ReLU fuction has faster convergence speed

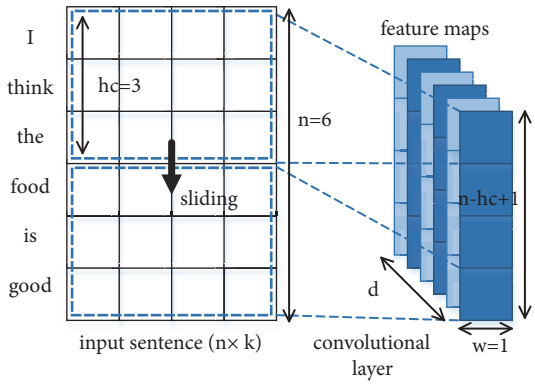


FIGURE 2: The convolution process with one input channel for an example sentence. Filter $W \epsilon R^{h_c \times k}$ extracts one feature $\epsilon R^{(n-h_c+1)\times 1}$ and with d filters we can extract d features.

of gradient descent. As a result of the convolution, a feature map c is generated where

$$c = \left[c_1, c_2, \ldots, c_{n-h_c+1}\right] \tag{3}$$

Figure 2 gives an example of conventional convolution. One feature is extracted from one filter. Assume the number of filters is d; then the shape of convolutional output is $(n - h_c + 1, 1, d)$. Intuitively, we can see that the width of output is compressed to 1, which means information loss.

3.3. Channel Transformation Based Convolutional Layer. To reduce information loss and take full advantage of information in depth dimension, namely, the channel information, we propose a method called channel transformation. Channel transformation is a reshape operation of a vector. For example, the shape of the convolution output is $(n - h_c + 1, 1, d)$, which can be transformed to $(n - h_c + 1, d, 1)$ by switch width and depth dimensions. Channel transformation provides two benefits. One is that the output of the previous convolutional can be remained "image" shape such that we can stack multiple convolutional layers. The other benefit is that we can make good use of information in depth dimension and extract feature maps with more sentimental semantics.

3.4. The Three CTB Convolutional Layers. In CTBCNN model, we have three CTB convolutional layers. Figure 3 shows the structure of the three channel transformation based convolutional layers.

In the first convolutional layer (conv1), the shape of the input vector is $(n, k, 1)$. Similar to Kim's work, we use 3 different sizes of filters.

$$\begin{aligned} &W_{11} \epsilon R^{h_{c11} * k}, \quad where\ h_{c11} = 3 \\ &W_{12} \epsilon R^{h_{c12} * k}, \quad where\ h_{c12} = 4 \\ &W_{13} \epsilon R^{h_{c13} * k}, \quad where\ h_{c13} = 5 \end{aligned} \tag{4}$$

Each size is with n_1 filters and extract n_1 feature maps. For filter W_{1i}, the shape of convolution output is $(n - h_{c1i} + 1, 1, n_1)$. The trick of channel transformation is used to switch width and depth dimensions such that we have the new shape $(n - h_{c1i} + 1,\ n_1, 1)$ as input for the next convolution layer.

In the second convolutional layer (conv2), we use 3 different filter sizes:

$$\begin{aligned} &W_{21} \epsilon R^{h_{c21} \times 1}, \quad where\ hc_{21} = n - h_{c11} + 1 \\ &W_{22} \epsilon R^{h_{c22} \times 1}, \quad where\ hc_{22} = n - h_{c12} + 1 \\ &W_{23} \epsilon R^{h_{c23} \times 1}, \quad where\ hc_{23} = n - h_{c13} + 1 \end{aligned} \tag{5}$$

Each size is with n_2 filters and extract n_2 feature maps. For filter W_{2i}, the shape of convolution output is $(1, n_1, n_2)$. The trick of channel transformation is used to switch height and depth dimensions to get the new shape $(n_2,\ n_1, 1)$. Finally we concatenate the output from 3 different sizes of filters on depth dimension and the shape of output becomes $(n_2,\ n_1, 3)$, which can be seen as an image of size $n_2 \times n_1$ with 3 channels.

In the third convolutional layer (conv3), we use filter $W_3 \epsilon R^{h_{c3} \times h_{c3}}$ to implement wide convolution, with n_3 filters and extract n_3 feature maps. Notice that the numbers of feature maps n_3 need to be the same as the output labels of the whole model. These feature maps contain sentimental semantics of the input sentence, which will be sent to global average pooling layer to capture the most important feature.

4. Global Average Pooling Layer

In classic CNN, feature maps produced by convolution layers are usually sent to max pooling layer to downsampling and then concatenated as a long vector, which is fully connected to output categories. The dense connection makes it hard to interpret how the category level information from the objective cost layer feed back to the convolution layer. Furthermore, the fully connected layers are prone to overfitting and heavily depend on dropout regularization.

For the reasons mentioned above, our method replaces the fully connected layer with global average pooling layer. Global average pooling layer enforces direct correspondences between feature maps and categories. In this way the feature maps can be intuitively interpreted as categories confidence maps. Furthermore, there is no parameter to optimize in global average pooling layer; thus overfitting is avoided.

From the comparison in Figure 4 we can see that global average pooling layer takes the average value of each feature map which can be regarded as confidence value for each category, and the resulting vector is fed directly into the Softmax function to get the probability distribution among sentiment categories.

$$P\left(y_j \mid x, \theta\right) = \frac{\exp\left(S_j\left(x, \theta\right)\right)}{\sum_{1 \le i \le |Y|} \exp\left(S_i\left(x, \theta\right)\right)} \quad 1 \le j \le |Y| \tag{6}$$

where θ represents the model parameter set. $S_j(x, \theta)$ is the average pooling result of the feature map that corresponds to category j. Y is the category space. We use stochastic gradient

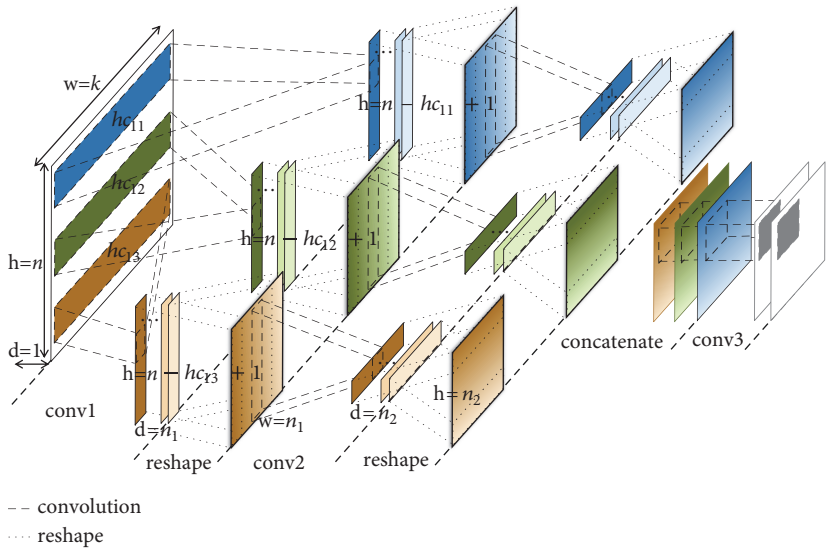
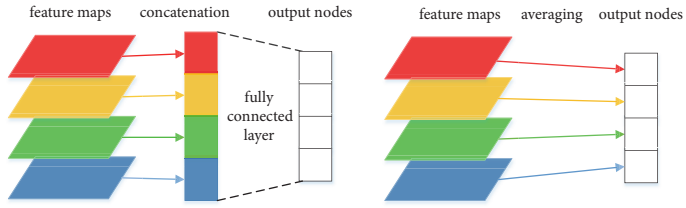


Figure 3: The structure of three channel transformation based convolutional layers. The first reshape switches the width and depth dimensions of the convolutional output. The second reshape switches the height and depth dimensions of the convolutional output.



(a) Fully connected layer (b) Global average pooling

Figure 4: The comparison between fully connected layer and global average pooling layer. Global average pooling layer feeds each feature map to corresponding category.

descent to minimize the Negative log-likelihood function of formula (6) and learn the parameter set.

5. Experiments

5.1. Overview. We have two parts of experiments. The first one performed Chinese sentiment classification tasks using CTBCNN, compared with other typical machine leaning methods and classic textCNN [15]. The second experiment focused on the regularization effect of global average pooling layer of CTBCNN model. At last we analyse the portability of our model and how it can be conveniently used in different cases.

All experiments are evaluated on two datasets: THU Hotel reviews (http://nlp.csai.tsinghua.edu.cn/~lj/.) used in [17] and ChnSentiCorp-Book (http://www.nlpir.org/?action-viewnews-itemid-77) used in [18]. Table 1 shows the details of these two datasets after duplication removal work. Both datasets are roughly equally split into positive and negative. Since there is no standard set for these dataset, we performed 10-fold cross validation and used the average accuracy metric to measure the overall performance for each method.

Initialized word vectors are 400-dimensional that were trained on 230000 articles from Chinese Wikipedia (https://dumps.wikimedia.org/zhwiki/latest/) using the skip-gram architecture. For model hyperparameters, we use 100 filters for conv1 and 100 filters for conv2. In conv3, the filter size is 3×3 with 2 filters which is in accordance with the number of categories.

Table 1: Summary of two datasets.

Dataset	Positive	Negative
THU Hotel reviews	7678	7811
ChnSentiCorp-Book	1967	1967

5.2. Sentiment Classification. To evaluate the performance of CTBCNN model, we experimented with several typical methods. All methods are experimented on two datasets with the same static word vectors. Results of our method against other methods are shown in Table 2.

The first line of Table 2 gives the results of lexicon-based baseline method used in [18, 19]. Experiment results show that most machine learning methods surpass the baseline method on both datasets. It is worth noticing that book reviews contain more less frequently used words and

Table 2: the classification results on two datasets.

Methods	Accuracy (%)	
	THU Hotel reviews	ChnSentiCorp-Book
Baseline(Lexicon-based)	80.00	71.30
CTBCNN	**92.43**	**92.81**
TextCNN	91.33	90.97
Naïve Bayes	75.26	83.20
Random Forest	80.68	84.98
SVM	89.11	86.63

Table 3: Global average pooling compared to fully connected layer.

Model	Accuracy (%)	
	THU Hotel reviews	ChnSentiCorp-Book
CTBCNN-FC without dropout	91.04	92.45
CTBCNN-FC with dropout	91.74	92.65
CTBCNN-GAP	**92.43**	**92.81**

expressions, which results in low classification accuracy. SVM outperformed other typical machine learning method, which shows that SVM is capable of handling high-dimensional features. Classic TextCNN achieves better performance than most machine learning method because of the powerful feature extraction ability of convolution layer, while CTBCNN surpasses all the other methods, which suggests that the CTB convolution layer can extract more representative feature than plain convolution layer.

5.3. The Regularization Effect of Global Average Pooling Layer. To evaluate the regularization effect of global average pooling layer, we set up a comparison experiment which replaces the global average pooling (GAP) layer of CTBCNN with a fully connected (FC) layer, while the other parts remain the same. We evaluate this model with and without dropout before the FC layer. All models are tested on the two datasets and the results are shown in Table 3.

As we can see in Table 3, for both datasets, the model with fully connected layer without dropout gives the worst performance, which is expected as the fully connected layer prone to overfitting [20] without applying any regularizer. The second model applies dropout before the fully connected layer and achieves better performance than the first one, while our model with global average pooling layer achieves the highest classification accuracy, which proved global average pooling is an effective way to avoid overfitting.

5.4. Portability of CTBCNN. From the above discussion we know that CTBCNN avoid human engineering by using CTB convolutional layer, which means CTBCNN is non-feature-specific and non-task-specific. This is helpful when different node handles different types of data in a distributed computing network. Besides, CTBCNN can be applied for multiclass classification tasks conveniently. All we need to do is make sure the number of the output feature maps of the last convolutional layer is equal to the number of categories.

6. Conclusion

Fog computing uses edge nodes to carry out a substantial amount of data analysis work. In public opinion analysis system, it is crucial to find suitable algorithm that is lightweight in operation and accurate in prediction. This work focuses on Chinese sentiment analysis in fog computing environment and proposes a non-task-specific method called Channel Transformation Based Convolutional Neural Network (CTBCNN). CTBCNN mainly consists of two parts: the three CTB convolutional layers and the global average pooling layer. CTB convolution layer is able to cover more channel information and extract more representative feature maps. And global average pooling is a regularizer to prevent overfitting. Through experiments and analysis, we show that our model do achieve competitive accuracy and it is convenient to apply this method to different cases in operation.

Data Availability

The "ChnSentiCorp-Book" data (also called "ChnSentiCorp-BK-ba-4000") used to support the findings of this study were supplied by Tan Songbo under license and so cannot be made freely available. Requests for access to these data should be made to Tan Songbo, tansongbo@software.ict.ac.cn. The "THU Hotel reviews" data used to support the findings of this study can be accessed from http://nlp.csai.tsinghua.edu.cn/~lj/.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Key R&D Program of China, Research and Application of Key Technologies

for Information Security Certification, under Grant no. 2016YFF0204001 of China Information Security Certification Center.

References

[1] D. C. Segura, R. de Souza Stabile, S. M. Bruschi, and P. S. Souza, "Providing computing services through mobile devices in a collaborative way - a fog computing case study," in *Proceedings of the 20th ACM International Conference on Modelling, Analysis and Simulation of Wireless and Mobile Systems*, pp. 117–121, ACM, Miami, FL, USA, November 2017.

[2] S. Park and Y. Yoo, "Network intelligence based on network state information for connected vehicles utilizing fog computing," *Mobile Information Systems*, vol. 2017, Article ID 7479267, 9 pages, 2017.

[3] D. Roca, R. Milito, M. Nemirovsky, and M. Valero, "Tackling IoT Ultra Large Scale Systems: fog computing in support of hierarchical emergent behaviors," in *Fog Computing in the Internet of Things*, pp. 33–48, Springer, Cham, Switzerland, 2018.

[4] S. Zhang, Z. Wei, Y. Wang, and T. Liao, "Sentiment analysis of Chinese micro-blog text based on extended sentiment dictionary," *Future Generation Computer Systems*, vol. 81, pp. 395–403, 2018.

[5] W. Xing, L. Haitao, and Z. Shaojian, "Sentiment analysis for Chinese text based on emotion degree lexicon and cognitive theories," *Journal of Shanghai Jiaotong University (Science)*, vol. 20, no. 1, pp. 1–6, 2015.

[6] B. Pang, L. Lee, and 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*, pp. 79–86, Association for Computational Linguistics, Stroudsburg, Pa, USA, July 2002.

[7] L. Shouhan, X. Rui, Z. Chengqing, and H. Churen, "A framework of feature selection methods for text categorization," 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*, pp. 692–700, Association for Computational Linguistics, 2009.

[8] W. Hongwei, Y. Pei, Y. Jiani, and L. JNK, "Text feature selection for sentiment classification of Chinese online reviews," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 25, no. 4, pp. 425–439, 2013.

[9] A. Severyn and A. Moschitti, "Twitter Sentiment Analysis with deep convolutional neural networks," in *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM, August 2015.

[10] L. Himabindu, R. Socher, and C. Manning, "Aspect specific sentiment analysis using hierarchical deep learning," *NIPS Workshop on Deep Learning and Representation Learning*, 2014.

[11] C. N. Dos Santos and M. Gatti, "Deep convolutional neural networks for sentiment analysis of short texts," in *Proceedings of the 25th International Conference on Computational Linguistics (COLING '14)*, 2014.

[12] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," https://arxiv.org/abs/1301.3781, 2013.

[13] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of the 27th Annual Conference on Neural Information Processing Systems (NIPS '13)*, pp. 3111–3119, December 2013.

[14] L. Min, C. Qiang, and Y. Shuicheng, "Network in network," https://arxiv.org/abs/1312.4400, 2013.

[15] Y. Kim, "Convolutional neural networks for sentence classification," https://arxiv.org/abs/1408.5882, 2014.

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, pp. 1097–1105, 2012.

[17] L. Jun and S. Maosong, "Experimental study on sentiment classification of Chinese review using machine learning techniques," in *Proceedings of the International Conference on Natural Language Processing and Knowledge Engineering, IEEE NLP-KE 2007*, pp. 393–400, Beijing, China, September 2007.

[18] Y. Zhang, X. Xiang, C. Yin, and L. Shang, "Parallel sentiment polarity classification method with substring feature reduction," in *Trends and Applications in Knowledge Discovery and Data Mining*, J. Li et al., Ed., vol. 7867 of *Lecture Notes in Computer Science*, pp. 121–132, Springer, Berlin, Germany, 2013.

[19] P. Zhang, Z. He, and L. Tao, "Compositional polarity classification approach for product reviews," in *Proceedings of the 2014 7th IEEE Joint International Information Technology and Artificial Intelligence Conference, ITAIC 2014*, pp. 58–62, Chongqing, China, December 2014.

[20] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.