Deep Sentiment Representation Based on CNN and LSTM

Qiongxia Huang
Faculty of Computer and Information Sciences
Fujian Agriculture and Forestry University
Fuzhou 350002, China
E-mail: jonesky_hqx@163.com

Xianghan Zheng
College of Mathematics and Computer Science
Fuzhou University
Fuzhou 350116, China
E-mail: xianghan.zheng@fzu.edu.cn

Riqing Chen*
Faculty of Computer and Information Sciences
Fujian Agriculture and Forestry University
Fuzhou 350002, China
E-mail: riqing.chen@fafu.edu.cn

Zhenxin Dong
Faculty of Computer and Information Sciences
Fujian Agriculture and Forestry University
Fuzhou 350002, China
E-mail: DongZhenxin2014@gmail.com

***Abstract*—Traditional machine learning techniques, including support vector machine (SVM), random walk, and so on, have been applied in various tasks of text sentiment analysis, which makes poor generalization ability in terms of complex classification problem. In recent years, deep learning has made a breakthrough in the research of Natural Language Processing. Convolutional neural network (CNN) and recurrent neural networks (RNNs) are two mainstream methods of deep learning in document and sentence modeling. In this paper, a model of capturing deep sentiment representation based on CNN and long short-term memory recurrent neural network (LSTM) is proposed. The model uses the pre-trained word vectors as input and employs CNN to gain significant local features of the text, then features are fed to two-layer LSTMs, which can extract context-dependent features and generate sentence representation for sentiment classification. We evaluate the proposed model by conducting a series of experiments on dataset. The experimental results show that the model we designed outperforms existing CNN, LSTM, CNN-LSTM (our implement of one-layer LSTM directly stacked on one-layer CNN) and SVM (support vector machine).**

Keywords-sentimentrepresentation, CNN, LSTM, sentiment classification

I. INTRODUCTION

With the growing development of social network information, massive text information that contains people's views, feelings, and comments has emerged on network mediums. The task of finding an effective approach of mining and analysis on these data is a very important research in the field of Natural Language Processing (NLP), which is called text sentiment analysis [1]. Text sentiment analysis mainly includes text classification, information extraction, and text generation techniques [2]. At present, sentiment analysis based on statistical machine learning has achieved good results in various applications [3]. However, the functions used in machine learning methods is simple, which may lead to their generalization ability to handle with the complex classification problem and the ability to express the complex function to be restricted to a certain extent under the condition of limited samples and computational units.

As one of key step in NLP, sentiment representation plays an important role in various tasks of emotion analysis. Traditional text processing methods use the bag-of-word model to generate the vector representation of the text [4]. However, the bag-of-word model suffers from the loss of grammar and semantics, which states that words are isolated without any contact. Better methods for sentence representation and document representation are currently based on distributed representation which was first proposed by Hinton in 1986 [5]. The main idea of distributed representation is to map words into a new space, and use the low dimensional real vectors to represent words, which are often called as "Word Representation" or "Word Embedding". This distributed representation can not only solve the problem of dimension disaster, but also find the correlation property of words, which improves the accuracy of vector semantics.

The invention of word embedding represents the significance of deep learning in NLP. With its emergence, the study of sentiment representation has made a breakthrough. In the current research trends of sentiment representation, there are two main kinds of models, which are tree-structured model and sequence-based model [6]. Sequence-based model gains deep semantic information by studying relationship between words that are in different positions of sequence. Tree-structured model learns syntactic information by converting text into syntactic parsing tree, each node in the tree corresponds to each word in text. Thus, deep learning can automatically learn deep sentiment representation on the basis of word embedding that can capture meaningful features.

Convolutional neural network (CNN) and recurrent neural networks (RNN) are the two widely used deep learning models for sentiment representation. CNN proposed

978-1-5386-2280-3/17 $31.00 © 2017 IEEE
DOI 10.1109/ICGI.2017.45

by LeCun in 1998 has strong adaptability and is extremely good at extracting local features from text [7]. It can significantly reduce the computational complexity as well as the number of training parameters due to its unique weight-sharing structure. For modeling sentences, Kalchbrenner *et al.* applied dynamic k-Max pooling in convolutional neural network and proposed the dynamic convolutional Neural Network, which is able to gain short and long-range relations [8]. As another popular network, RNNs can deal with sequence data and learn long-term dependence [9]. RNNs take into account the relationship between the current output and the previous output of sequence, which means the input of current hidden layer not only includes the output of the input layer but also includes the output of the previous hidden layer. Besides, LSTM (Long Short-term Memory RNNs) was designed to gain better expression of long and short time dependence and handle with the problem of gradient diffusion and gradient explosion caused by standard RNNs [10]. In this paper we design a model for deep sentiment representation based on CNN and LSTM, which performances as a simple end-to-end structure. A series of experiments have been conducted in sentiment classification to evaluate deep sentiment representation learned from the network we design.

II. Related Work

With the rapid growing of study about word embedding, a series of methods have been designed for the representation of sentences/documents through semantic composition. CNN, RNNs, LSTM, and some improved models on the basis of them are strong neural network for sentiment representation. CNN owns convolution layer and pooling layer, such unique network makes it easy to automatically extract local features and reduce the complexity. Kim *et al.* proposed a simple network including one-layer convolution operation with filters using different width, and a max pooling layer, which performed successfully in sentiment classification after adding one fully connected layer finally [11]. Unlike the model proposed by Kim *et al.*, the Johnson *et al.* employed bag-of-word model to represent text instead of low-dimensional word vectors, which are proved to be extremely effective in text categorization [12].

Many researchers are also devoted to the research of RNNs, LSTM, and recursive neural network due to their ability of processing sequence data and extracting syntax features. Combining RNNs and autoencoder, Kyunghyun *et al.* designed a new model which is named RNN Encoder-Decoder, and it shows good ability of learning meaningful representation when applied in statistical machine translation [13]. As a slight variant of recursive neural network, Socher *et al.* used vector as well as matrix to deal with each node of the tree structure in recursive neural network, which successful captures sentiment representation for phrases and sentences [14]. In addition, Tai *et al.* reformed the recursive neural network by using the cell block of LSTM model to represent every non-leaf node of the tree structure in the network and designed a new model to improve semantic representation [15].

III. The Proposed Approach

In this paper, a new approach based CNN and LSTM are proposed in order to capture deep sentiment features. The network structure of the approach we designed is shown in Fig. 1, which has one-layer CNN and two-layer LSTMs that are stacked in order. Blocks with the same color correspond to different convolution operations of various filters. The final output of the model is the last hidden unit of the second-layer LSTM. In order to simplify the description, we use the English word as input in the Fig. 1.

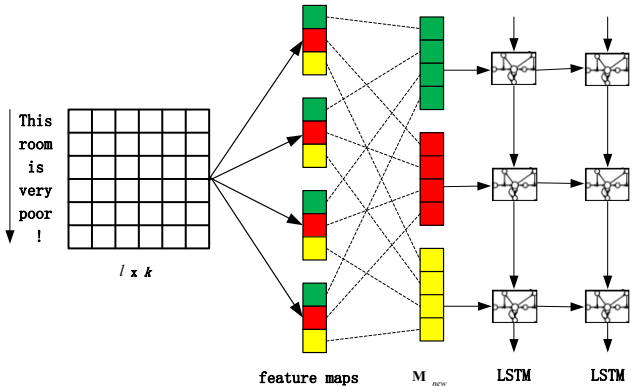

Fig. 1.The architecture of our model for sentence modeling.

A. Convolution for Extracting N-gram Features

The word2vec toolkit provided by Google is used to generate word vectors, and then the pre-trained word vectors are connected in series to represent each sentence. Let $\mathbf{x}_i \in \mathbb{R}^k$ be the k-dimensional word vector of the $i-$th word in a sentence. Let l be the length of a sentence, so the input sentence can be denoted by $\mathbf{x} \in \mathbf{R}^{l \times k}$, where $\mathbf{X}$ is represented as

$$\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots \mathbf{x}_l\} \tag{1}$$

Here, the number of filters in one-layer convolution neural network is denoted as n, and the size of each filter is set as s. Let $\mathbf{w} \in \mathbf{R}^{sk}$ stands for a filter involved in convolution operation which applied in a window composed of words to extract n-gram feature, therefore a feature generated by convolution operation can be denoted as

$$\mathbf{m}_j = f(\mathbf{w} \circ \mathbf{x}_{i:i+s-1} + b) \tag{2}$$

Where $\circ$ means element-wise multiplication, b is a bias term, and f stands for the nonlinear function, and $\mathbf{x}_{i:i+s-1}$ is represented as

$$\mathbf{x}_{i:i+s-1} = \mathbf{x}_i \oplus \mathbf{x}_{i+1} \cdots \oplus \mathbf{x}_{i+s-1} \tag{3}$$

Where $\oplus$ means the concatenation of row vectors, after the filter glides through the whole sentence with the form of

a window, a feature map will be produced and it is denoted as

$$\mathbf{m} = [m_1, m_2, \cdots m_{l-s+1}], \quad \mathbf{m} \in \mathbf{R}^{l-s+1} \tag{4}$$

We just introduce the process of one feature map that is captured from one filter. The one-layer CNN model has n filters and will produce multiple feature maps which are represented as

$$\mathbf{M} = \{\mathbf{m}_1; \mathbf{m}_2; \cdots; \mathbf{m}_n\} \tag{5}$$

Where $\mathbf{M} \in \mathbf{R}^{n \times (l-s+1)}$ and semicolons means row vector concatenation and. In order to preserve sequentiality of the original sentences and make the LSTM model learn long-term dependencies more accurately, we use $\mathbf{M}_{new}$ as the input of the two-layer LSTMs, where $\mathbf{M}_{new}$ is produced as follows:

$$\mathbf{M}_{new} = \mathbf{M}^T \tag{6}$$

Where T means matrix transpose.

B. Capturing Long-term Dependencies

The LSTM model has been proved to be extremely effective in capturing long-distance correlations in sequence with arbitrary length [16]. In our model, we adopt two-layer LSTMs stacked on CNN, each LSTM is corresponds to the version designed by Zaremba and Sutskever [17].

LSTM processes the input vectors by recursive execution of cell block which is dependent on the old hidden state h_{t-1} as well as the current input x_t, where t means the current time and $t-1$ refers to the former time. In cell block, let i_t be the input gate, let f_t stands for the forget gate, let o_t be the output gate, let c_t represents the current memory cell, so that the operational principle of LSTM can be described as follow

$$\begin{aligned} i_t &= \sigma(W^{(i)}x_t + U^{(i)}h_{t-1} + b^{(i)}) \\ f_t &= \sigma(W^{(f)}x_t + U^{(f)}h_{t-1} + b^{(f)}) \\ o_t &= \sigma(W^{(o)}x_t + U^{(o)}h_{t-1} + b^{(o)}) \\ u_t &= \tanh(W^{(u)}x_t + U^{(u)}h_{t-1} + b^{(u)}) \\ c_t &= i_t \circ u_t + f_t \circ c_{t-1} \\ h_t &= o_t \circ \tanh(c_t) \end{aligned} \tag{7}$$

Where σ refers to the logistic sigmoid function, $W^{(i)}$ and $U^{(i)}$ stand for weights to learn during the process of the model, the $\circ$ means the element-wise multiplication. In addition, the value of f_t, i_t and o_t are in [0, 1]. The all hidden state vectors of the first-layer LSTM are as the input of the second-layer LSTM, then the hidden state vector at the last time step of which are the deep representation for sentence.

C. Sentiment Classification

In this section, the learned deep representation for sentence is fed to a sigmoid layer for text classification. We choose the cross-entropy as loss function and employ the optimizer RMSprop to train the model. The cross-entropy error are represented as

$$J(W) = -\sum_{i=1}^{c} y_i \ln a_i \tag{8}$$

Where y_i refers to the true label of the $i-$th sample, a_i is the final label that model predicts for the $i-$th sample, and the notation $\sum$ means summation.

IV. Prepare Evaluation

We apply the designed model for deep representation based on CNN and LSTM in sentiment classification to evaluate its effectiveness. In this section, we describe the experiment setup, experiment results, and model analysis in detail.

TABLE I
ACCURACY AND F1 VALUE OF FIVE MODELS ON D_1

Model	Accuracy	F1
CNN	85.6%	0.854
LSTM	83.8%	0.834
CNN-LSTM	84.2%	0.829
SVM	86.0%	0.860
Our Model	87.2%	0.868

TABLE II
TRAINING TIMES OF OUR MODEL ON CPU AND GPU WITH D_1

Processor Unit	Training Times
CPU	145*s*
GPU	17*s*

A. Datesets and Experiment Setup

We use one dataset to conduct a series of experiments. The dataset is the comments about sina micro-blog provided by the seventh COAE (Chinese Orientation Analysis and Evaluation) conference which are denoted as D_1. Each instance of the D_1 is labeled with sentiment polarity that is either positive or negative. The dataset is in Chinese language and its percentages of positive and negative are all 50%. The D_1 contains 7226 comments, 7/8 of which are training sets and 1/8 of which are test sets.

The model we designed is compared with four methods, including CNN, LSTM, CNN-LSTM, and SVM. We also use word2vec toolkit to produce word vectors and make the concatenation of them as the input of CNN, LSTM, and

CNN-LSTM. Different with them, the input of SVM is processed by bag-of-word model. The accuracy and F1 value are used to evaluate the performance of the model. Our model is implemented on Keras, which emerges as a popular python library. We train the models on CPU and GPU respectively and compare their consumption time. For hyper parameters setting, the number of filters with length of 3 is set to 150, the memory dimension of LSTM is set to 128, the probability of dropout layer is set to 0.5 and the factor of Gaussian noise is set to 0.01.

B. Experiment Results

The results of the proposed model and other four models on the D_1 are show in Table I. It can be see that the accuracy of the proposed model on the D_1 is 87.2%, which is 4 percentage points higher than that of LSTM. And the F1 value of the proposed model on the D_1 is 0.868, which is 4 percentage points higher than that of CNN-LSTM. It is clear that our model achieve the best performance on D_1.

We can find that using CNN alone or using LSTM alone is unable to achieve the desired results, this is because CNN fails to learn long-term dependencies while LSTM is unable to capture local features. Although combining the advantages of CNN and LSTM, CNN-LSTM does not obtain the optimistic results on the D_1, because the output of CNN is directly sent to LSTM without any process, which is unable to maintain the sequentiality of the original data. What's more, one-layer LSTM is not enough for extracting deep long-distance correlations in a sequence. Besides, as show in Table II, the unit *s* refers to seconds, due to the ability of GPU to accelerate calculations, training time of our model on CPU takes about ten times as much as that of our model on GPU. In concluding, it is clear that our model achieves good results in extracting deep sentiment representation for sentiment classification due to its special network structure.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we have introduced a novel model for deep sentiment representation by skillfully combining one-layer CNN and two-layer LSTMs. Due to convolution layer and pooling layer, the CNN makes it easy to automatically extract local features and reduce the computation complexity. Besides, LSTM is skilled in learning sequence characteristics due to its ability to learn syntax features of linguistics. Therefore, our model achieves good performance on sentiment classification. In the future work, we may study the combination of the bag-of-word model and word embedding technology to produce better pre-training feature as the input of network, and meanwhile change the way that we combine CNN and LSTM in our model.

REFERENCES

[1] B. Pang and L. Lee , "Opinion mining and sentiment analysis", Foundations and Trends® in Information Retrieval, vol. 2, no. 1-2, pp. 1-135, July. 2008.

[2] B. Liu,"Sentiment analysis and opinion mining", Synthesis lectures on human language technologies, vol. 5, no. 1, pp. 1-167, May. 2012.

[3] M. S. Neethu, and R. Rajasree,"Sentiment analysis in twitter using machine learning techniques", Computing Communications and Networking Technologies (ICCCNT), 2013, pp. 1-5.

[4] A. Agarwal, B. Xie, I. Vovsha, O. Rambow, and R. Passonneau, "Sentiment analysis of twitter data", in Proc.Association for Computational Linguistics, 2011, pp. 30-38.

[5] G. E. Hinton,"Learning distributed representations of concepts", Proceedings of the eighth annual conference of the cognitive science society, 1986, pp. 1-12.

[6] C. Zhou, C. Sun,Z. Liu, and F. Lau, "A C-LSTM neural network for text classification", 2015, arXiv preprint arXiv :1511.08630.

[7] Y. LeCun, L. Bottou, and Y. Bengio, "Gradient-based learning applied to document recognition", Proceedings of the IEEE, 1998, pp.2278-2324.

[8] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences", 2014, arXiv preprint arXiv :1404.2188.

[9] T. Mikolov, and M. Karafiát, L. Burget, and J. Cernocky, "Recurrent neural network based language model", in Interspeech, vol. 2, pp. 3, 2010.

[10] S. Hochreiter, and J. Schmidhuber, "Long short-term memory", Neural computation, vol. 9, no. 8, pp. 1735-1780, Nov. 1997.

[11] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model", Journal of machine learning research, vol. 3, pp. 1137-1155, Feb. 2003.

[12] T. Mikolov, I. Sutskever, K. Chen, and G.S. Corrado, "Distributed representations of words and phrases and their compositionality", in Advances in neural information processing systems, 2015, pp. 3111-3119.

[13] Y. Kim, "Convolutional neural networks for sentence classification", 2014, arXiv preprint arXiv :1408.5882.

[14] R. Socher, B. Huval, C. D. Manning, and A. Y.Ng, "Semantic compositionality through recursive matrix-vector spaces", Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. Association for Computational Linguistics, 2012, pp. 1201-1211.

[15] K. S. Tai, R. Socher, and C. D. Manning., "Improved semantic representations from tree-structured long short-term memory networks", 2015, arXiv preprint arXiv :1503.00075.

[16] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions", International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, vol. 6, no. 2, pp. 107-116, Apr. 1998.

[17] W. Zaremba, and I. Sutskever, "Learning to execute", 2014, arXiv preprint arXiv :1410.4615.