Character-level Neural Networks for Short Text Classification

Jingxue Liu
School of Computer Science and Technology,CUMT
XuZhou, JiangSu, China
liujx@cumt.edu.cn

Fanrong Meng
School of Computer Science and Technology,CUMT
XuZhou, JiangSu, China
mengfr@cumt.edu.cn

Yong Zhou
School of Computer Science and Technology,CUMT
XuZhou, JiangSu, China
yzhou@cumt.edu.cn

Bing Liu
School of Computer Science and Technology,CUMT
XuZhou, JiangSu, China
liubing@cumt.edu.cn

Abstract—*Since short text is characterized of the short length, sparse features and strong context dependency, the traditional models have a limited precision. Motivated by this, this article offers an empirical exploration on a character-level model which implements a combination of convolutional neural network(CNN) and recurrent neural networks(RNN) for short text classification. Including the highway networks framework so that we can address the difficult of training and improve the accuracy of classification. The evaluations on several datasets showed that our proposed model outperforms the standard CNN and traditional models on short text classification mission.*

Keywords—character-level; CNN; text classification

I. INTRODUCTION

With the rapid development of internet technology and mobile social network, the internet information such as real-time news, film reviews and community notes which is usually seen as short text has grown rapidly. Those texts are short, context-dependent and have very important research value in several natural language processing(NLP) tasks such as information retrieval, text recommendation and relation extraction. As the foundation of the subsequent text-based processing, the short text classification is usually seen as a sentence-level classification problem. In addition, in the process of smart cities construction, classification technology is widely used in various scenes, such as the news classification, opinions bias analysis, spam screening and sentiment analysis. Especially in the era of big data, massive text data's processing is also a great challenge for all researchers. So convincing efficiency and accuracy of the language model for short text classification is being paid more and more by researchers.

For a long time, some classical machine learning methods such as Naive Bayes, k-Nearest Neighbor(KNN), Support Vector Machine (SVM) and conditional random fields, are the main streams to processing text classification. But short text is characterized of the short length, sparse features and strong context dependency, the traditional models have a limited precision. Recently, with the great success of deep and neural learning in computer vision and speech processing tasks, deep learning has been applied to many tasks of NLP. The deep neural networks can improve the accuracy of the classification by combining the low-level text information and forming more abstract high-level text representation through the multi-layer networks. For example, convolution neural network (CNN) has excellent feature self-extraction ability, which can significantly reduce the difficulty of manually extracting features in text classification. And the word embedding technique based on the neural network model provides a new idea for semantic vector representation of short text. In this manner, each sentence can be treated as a matrix. This mapping method inspires researchers to apply convolutional neural networks (CNN), one of the common architectures of deep learning, to sentence classification. Thus, many CNN-based methods were proposed and they can achieve satisfactory performance as expected. In all CNN-based methods, CNN-non-static, a single-layer and single-channel sentence model proposed by Y.Kim, is the simplest method and has quite satisfactory performance in [1].

To date, almost all techniques of text classification are based on words, as the same question, the traditional models basing on word embedding still have a limited precision in dealing with massive web text data. Especially, working on words also has the trouble with abnormal character combinations such as misspellings and emoticons may be naturally learned.

Motivated by solving the above problems, this article offers an empirical exploration on a neural networks model which works on only characters for short text classification mission. The results of the experiment executed on 3 tasks show that our methods can achieve better performance than many other complex CNN models.

To summarize, our contributions are as follows:

- We implement a model work on only characters, which achieves fewer parameters on several datasets and
- including the highway networks framework addresses the training difficult and improve the accuracy of classification.

978-1-5386-2524-8/17/$31.00 ©2017 IEEE

In remaining parts, section 2 presents the background knowledge of related work. Section 3 gives the details of our models. Section 4 reports the experiment. And our work is concluded in section 5.

II. BACKGROUND

A. Related Work

As we all know, it has become a fundamental component of deep learning in NLP since Google's Tomas Mikolov made Word2Vec in [2]. The basic idea of Word2Vec is to replace each word in the natural language as a short vector of uniform mean and dimension. To be simple, Word2Vec's training model cloud be seen as a hidden layer of the neural network, its input and output are both the vector of vocabulary. So this mapping method inspired us to apply convolutional neural networks (CNN) to short text classification, and lots of CNN-based methods followed.

But many researchers have found CNN are more useful in extracting information from raw signals, which means character level embedding also has great research space. Here are some relevant works at the character level are as follows: Santos used the English short text character sequence as a processing unit to learn the word and sentence-level features of the text, respectively, to improve the accuracy of short text classification in [3].Johnson reduced the model's learning parameters by directly using the one-hot vector as the input of the convolution neural network in [4].Zhang explored treating text as a kind of raw signal at character level, and applying temporal(one-dimensional) ConvNets to it in [5].And he implemented a model of character-level convolutional networks for text classification. These approaches have been proven to be competitive to traditional models.

B. Character Embedding

Word2vec is a particularly efficient predictive model for learning word embeddings from raw text. It comes in two flavors, the Continuous Bag-of-Words model(CBOW) and the Skip-Gram model. Word2vec is a particularly efficient predictive model for learning word embeddings from raw text. It comes in two flavors(see fig-1), the Continuous Bag-of-Words model(CBOW) and the Skip-Gram model. CBOW and skip-gram cloud be considered one of the core concepts of word2vec. In fact, those two models have a lot of similarity, so here we mainly introduce the CBOW model, then show some differences between skip-gram and CBOW.

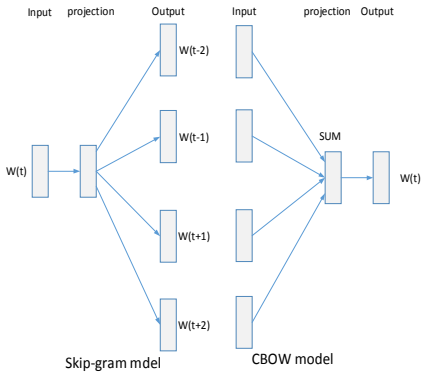

Figure 1: The Skip-gram and CBOW model architecture

Both the CBOW and the skip-gram are based on the Huffman tree. It is worth to note that the initial value of the intermediate vector stored by the non-leaf nodes in the Huffman tree is zero, and the word vector of the word corresponding to the leaf node is randomly initialized. The training process as shown below, there are input, projection and output three stages. For an example, a target word (e.g. 'cat'), the input layer is the word vector of n-1 words around the word cat. So the CBOW predicts target words from source context words ('the cat sits on the table'). And compared with CBOW, only minor differences, the input to the skip-gram is the word vector of the current word, and the output is the word vector of the surrounding word.

In this article, the character level embedding also draws on the Word2Vec word embeddings from Mikolov, each character will be mapped to a vector of pre-set dimensions. The detail of character-level embeddings for CNN shown in Figure 2.

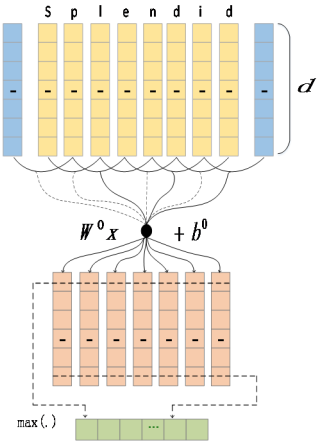

Figure 2: Convolutional approach to character-level feature extraction.

III. MODEL ARCHITECTURE

In this section, we introduce the design of character-level deep learning model for text classification. The design is modular, where the gradients are obtained by back-propagation to perform optimization. Our model employs a convolutional neural network (CNN) and a highway network over characters, whose output is given to a long short-term memory(LSTM) recurrent neural network model. The architecture of our model, shown in Figure 3.

For notation, we denote vectors with bold lower-case (e.g.$\mathbf{x_t}$,$\mathbf{b}$), matrices with bold upper-case (e.g.$\mathbf{W}$, $\boldsymbol{U^0}$), scalars with italic lower-case (e.g.x, b), and sets with italic upper-case (e.g.V, C) letters.

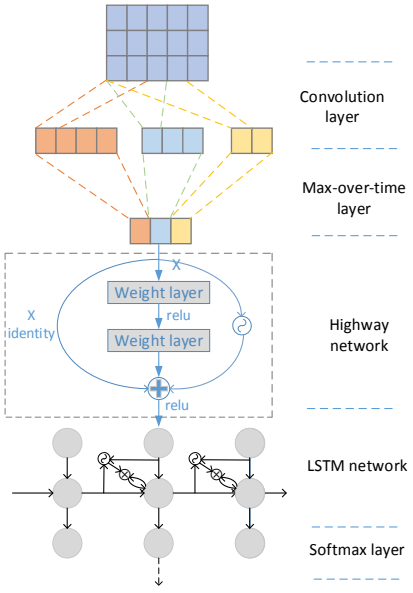

Figure 3: Our model architecture. First layer performs a lookup of character embeddings and stacks them to form the matrix $\boldsymbol{A}$ .Then convolution operations are applied between $\mathbf{A}$ and multiple filter matrices. A max-over-time pooling operation is applied to obtain a fixed-dimensional representation of the word, which is given to the highway network. The highway network's output is used as the input to a multi-layer LSTM. Finally, an affine transformation followed by a softmax is applied over the hidden representation of the LSTM to obtain the distribution over the next word. Cross entropy loss between the (predicted) distribution over next word and the actual next word is minimized. Element-wise addition, multi-plication, and sigmoid operators are depicted in circles, and affine transformations are represented by solid arrows.

A. Character-level Convolutional Neural Network

Convolution neural network(CNN) has excellent feature self-extraction ability, which can significantly reduce the difficulty of manually extracting features in text classification. So our methods select the CNN-non-static architecture which is a single-layer and single-channel CNN-based sentence model proposed by Kim.

In the CNN-non-static, each character in one sentence will be saved including abnormal character combinations such as misspellings and emoticons in the pretreatment stage, so that we can make sure the integrity of the text information. Using the technique of character-level embeddings, each character will be replaced with its vector representation, thus the short text generates a matrix $\mathbf{A} \in \boldsymbol{R}^{s \times d}$ where s is its length with zero padding and d is the dimensionality of the character embeddings. And then $\mathbf{A}$ is convolved with some linear filters to get some convolved feature vectors $\mathbf{v}$.Each filter window is defined as a matrix $\boldsymbol{W_i} \in \boldsymbol{R}^{h \times d}$ where h denotes the number of words involved in each operation. And k-max pooling operation is applied to extract largest element from each convolved feature vector $\mathbf{v}$.All largest elements finally constitute a CNN feature vector $\mathbf{c} \in \boldsymbol{R}^{k}$where k is the number of filters.

The above is the classic CNN-based approach of feature extraction, but we don't choose to directly fed our feature vector to fully connected layer with classifier. Instead, We apply RNN to improve the accuracy of classification for the advantages of modeling sequential phenomena.

B. LSTM Model

As for the recurrent neural network(RNN), long short-term memory(LSTM) addresses the problem of learning long range dependencies by augmenting the RNN with a memory cell vector at each time step, Concretely, one step of an LSTM takes as input $\boldsymbol{x_{t-1}}, \boldsymbol{h_{t-1}}, \boldsymbol{c_{t-1}}$ and produces $\boldsymbol{h_t}, \boldsymbol{c_t}$ via the following intermediate calculations:

Input Gate: Controls how much of the current input x_t and the previous output h_{t-1}will enter into the new cell.

$$i_t = \sigma(W^i x_t + U^i h_{t-1} + b^i) \quad (1)$$

Forget Gate: Decide whether to erase (set to zero) or keep individual components of the memory.

$$f_t = \sigma(W^f x_t + U^f h_{t-1} + b^f) \quad (2)$$

Output Gate: Scales the output from the cell.

$$o_t = \sigma(W^o x_t + U^o h_{t-1} + b^o) \quad (3)$$

Cell Update: Transforms the input and previous state to be taken into account into the current state.

$$g_t = tanh(W^g x_t + U^g h_{t-1} + b^g) \quad (4)$$

Internal State update:Computes the current timestep's state using the gated previous state and the gated input.

$$c_t = f_t . c_{t-1} + i_t . g_t \quad (5)$$

Hidden Layer: Output of the LSTM scaled by a tanh (squashed) transformations of the current state.

$$h_t = o_t . tanh(c_t) \quad (6)$$

Here $\sigma(\cdot)$ and $\tanh(\cdot)$ are the element-wise sigmoid and hyperbolic tangent functions, $\cdot$ is the element-wise multiplication operator, and $\boldsymbol{i_t}, \boldsymbol{f_t}, \boldsymbol{o_t}$ are referred to as input, forget, and output gates. At $t = 1$, $\boldsymbol{h_0}$and $\boldsymbol{c_0}$ are initialized to zero vectors. Parameters of the LSTM are $\boldsymbol{W^j}, \boldsymbol{U^j}, \boldsymbol{b^j}$ for $j \in \{i, f, o, g\}$ so it is easy to extend the RNN/LSTM to our category prediction model. And in practical application, having multiple layers is often crucial for obtaining competitive performance on various tasks.

Our model get the output from a character-level convolutional neural network as the input embeddings $\mathbf{X}$.Similarly, for our model which usually takes words as inputs, if $w_t = k$, then the input to the model at t is the input embedding $\boldsymbol{X^k}$,the k^{th}column of the embedding matrix $\mathbf{X} \in \boldsymbol{R}^{n \times |V|}$. As for the given historical sequence, which comes from the LSTM layers' output, follows by a softmax:

$$\Pr(w_{t+1} = j \mid w_{1:t}) = \frac{\exp(h_t \cdot p^j + q^j)}{\sum_{j' \in V} \exp(h_t \cdot p^{j'} + q^{j'})} \quad (7)$$

Where V is the fixed size vocabulary of words $w_{1:t} = [w_1, \ldots, w_t]$ is the given the historical sequence. p^j is the j^{th} column of $\mathbf{P} \in \mathbf{R}^{\mathbf{m} \times |\mathbf{V}|}$ (also referred to as the output embedding),and q^j is a bias term. After the above process, our feature vector is fed to LSTM model with dropout and goes through a softmax function to get a classification probabilistic vector $\mathbf{P} \in \mathbf{R}^{\mathbf{n}}$where n is the number of labels.

C. Highway Network

We could simply replace $\mathbf{x}^{\mathbf{k}}$ with $\mathbf{y}^{\mathbf{k}}$ at each t in the

RNN, and as we show later, this simple model performs well on its own (Table II), but it's not good enough for text classification. Instead, we implement a new architecture designed to ease gradient-based training of very deep networks—highway networks, proposed by Srivastava in [6], since they allow unimpeded information flow across several layers on information highways.

The architecture is characterized by the use of gating units which learn to regulate the flow of information through a network. Whereas the highway neural network typically consists of L layers where the l^{th} layer $(l \in \{1,2,\ldots L\})$ applies a non-linear transform H on its input $\mathbf{x_i}$ to produce its output $\mathbf{y_i}$.

$$y = H(x, W_H) \tag{8}$$

H is usually an affine transform followed by a non-linear activation function, but in general it may take other forms. For every layer of highway network, it does the following:

$$y = H(x, W_H) \cdot T(x, W_T) + x \cdot (1 - T(x, W_C)) \tag{9}$$

We refer to T as the transform gate and (1-T)as the carry gate, since they express how much of the output is produced by transforming the input and carrying it, respectively. The dimensionality of $\mathbf{x}, \mathbf{y}, \mathbf{H}(\mathbf{x}, \mathbf{W_H})$ and $\mathbf{T}(\mathbf{x}, \mathbf{W_T})$ must be the same for Equation (2) to be valid.

Thus, depending on the output of the transform gates, a highway layer can smoothly vary its behavior between that of a plain layer and that of a layer which simply passes its inputs through.

IV. EXPERIMENTS

In order to verify the validity of our model, this paper compares experiments on several selected datasets. The English data sets come from the original article[7].We also collected some of the Chinese film comments on the DouBan website, so that we can see the effect of our character-only model dealing with the abnormal character combinations. The environment of our work is basing on TensorFlow(1.1.0rc1) deep learning platform.

A. Datasets

- **MR**: movie reviews dataset. There is one sentence per review, and the task is to classify positive/negative reviews.
- **SST-1**: Stanford Sentiment Treebank with fined-rained labels(very positive, positive, neutral, negative, very negative).It is an extension of MR and has train/dev/test splits.
- **SST-2**:Same as SST-1but with neutral reviews removed and binary labels.
- **Irony**:This contains 16,006 sentences from reddit labeled as ronic.The dataset is imbalanced(relatively few sentences are ironic).Thus before training, we under-sampled negative instances to make classes sizes equal.
- **Tweet**: Tweets from 10 different authors.Classification involves classifying which belongs to which author.
- **DR**: Chinese movie reviews dataset similar as MR. This dataset was not used in the original article.

TABLE I: Summary statistics for the datasets after tokenization.

Data	c	l	m	N	$\lvert V \rvert$	Test
MR	2	20	56	10662	18765	CV
SST-1	5	18	53	11855	17836	2210
SST-2	2	19	53	9613	16185	1821
Irony	2	75	874	1074	6138	CV
Tweet	10	39	62	25552	33438	5964
DR	2	67	2565	35124	4602	CV

B. Experimental Setup

1)Hyperparameters and Training: In our implementation of the model we experimented with a lot of different configurations. Below is a list of parameters and specifications that were used for all experiments:

- **Character embeddings**: We used the pre-trained Word2Vec mentioned earlier. Each character embedding is in < 300. For characters that are not found in Word2Vec we randomly initialized them with a uniform distribution in the range of [-0.5, 0.5].
- **Filters**: We used filters with window sizes of [1,2,3,4,5,6,8] and the number of each filter is [50,100,150,150,200,200,200] For activation we used ReLU.
- **Dropout rate**: 0.5.
- **Mini-Batch size**: 32.
- **Highway layers**: 3.
- **RNN hidden_unit**: 300.
- **RNN layer_depth**:2.
- **Optimizer**: We decided to use the more recent ADAM optimizer,as it seemed to converge much faster and in some cases improved the results.
- **Learning rate**: 0.001.
- **Number of epochs**: For Tweet where we used 15, MR and SST-1 where we used 10 and irony where we used 5.For the large DR detset, we used different number of epochs, you will see in the experimental results and discussion.
- **training**: Evaluate the results on dev set and save the best model for every 100 step. For irony, we evaluate the results on dev set and save the best model for every 10 step.
- **l2-loss**: We added l2-loss with λ = 0.0 on the weights and biases of the final layer.

- **Others**: For datasets without a standard dev set we use 10-fold cross validation, which randomly select 10% of the training data as the dev set.

2)Evaluation standard: In our work the evaluation of the classification is mainly measured by the accuracy. For a example, if we have a data set of size N ,our inputs is x_i and its label y_i,outputs will be $\hat{y}_i$ .So

$$Accuracy = \frac{\sum_{i=1}^{N} | y_i = \hat{y}_i |}{N} \quad (10)$$

C. Experimental Results and discussion

This paper compares multiple models, including some traditional method. The traditional classification method selected NB, KNN and Linear-SVM three models, convolution neural network used the CNN-static, CNN-static, CNN-non-static and CNN-multichannel, which CNN-static and CNN-non-static are classified into CF or PF according to the input level.

1)Our method vs Traditional methods: in comparison with other methods, our methods have obvious advantage and achieve best results on several tasks. Basing on the DR data set, we selected 10,000 positive reviews and 10,000 negative comments respectively as small DR data set. We can see a comparison between our results and the ones in the original article in [30]. The experiment results are listed in table II.

TABLE II: Summary statistics for the results of traditional methods.

Model	Accuracy
NB	72.97
KNN	71.73
Linear-SVM	76.24
CNN-rand	81.34
CNN-static	84.41
CNN-non-static	84.32
CNN-multi-channel	84.27
CNN-char-static	81.47
CNN-char-non-static	81.69
Our work	87.25

By comparing the results of the CNN-based model with the traditional classification model, the CNN-based model works better. Our work has already achieved the best performance,which only includes three hidden layers of highway network. Because our complete model adds LSTM just to improve the effect of classification, the accuracy of the complete model will be further improved,which will be showed in the following experiment. And it can be seen from Table 2, the effect of CNN-rand model is similar to that of CNN-chr-static model and CNN-chr-non-static model, and is superior to NB, KNN and Linear-SVM. We also use the pre-trained Word2Vec in the method of drawing above, we found that if the CNN-based model use the pre-trained Word2Vec works better. From this point of view, CNN-based model of the characteristics of automatic extraction ability, it is worth further study in this article.

2)Have highway networks vs None: we also validate the effect of the highway networks in our model, basing on the large DR data set, we selected 18,576 positive reviews and 17,548 negative comments respectively. The experiment results are listed in table IV.

TABLE III: Results on the original article datasets that were tested in our model

Model	MR		SST-1		SST-2		Irony		Tweet	
	Orig	***Ours***	***Orig***	***Ours***	***Orig***	***Ours***	***Orig***	***Ours***	***Orig***	***Ours***
CNN-rand	76.4	74.6	41.0	45.0	80.2	82.6	60.2	67.1	89.1	84.2
CNN-static	80.3	82.1	48.1	46.2	85.4	84.5	60.5	68.7	83.8	80.8
CNN-non-static	80.5	81.2	47.3	48.0	85.8	85.0	62.1	68.7	89.2	85.3

TABLE IV: Summary statistics for the results of our model with different highway layers.

Model	Accuracy
CNN-char-non-static	85.25
CNN+1highwaylayer	92.94
CNN+2highwaylayer	94.68
CNN+3highwaylayer	95.36
CNN+RNN	94.96
CNN+highway+RNN	95.56

In general, the difficulty of deep network training is due to the problem of gradient return obstruction, so the highway networks add a gate function that allows the output of the network to consist of the direct input of the network. Look at the drawing above, respectively, there is no highway network and with different hidden layers. With the increase of hidden layers in highway network, the problem of deep network training has been alleviated.

3) Our method vs other methods: in comparison with other methods, our methods also have obvious advantage on several tasks. The experiment results are listed in table III.In this series of experiments,we all chose the complete model,which means we added the highway layers and the highway network's output is used as the input to a multi-layer LSTM. For the

datasets, we also chose some of them used in the original article.

In table III above, we can see a comparison between our results and the ones in the original article in [7]. For the CNN in our methods, we use simplest architecture which applies only one set of character-level embeddings as input and has only one convolution layer. We can see that overall, our results are comparable (and sometimes better) to the ones in the original article. Due to the different data sets, the experimental results are also affected by parameter settings. In the above experimental setup section, we only listed the parameters' benchmark, in the actual experiment there are subtle parameter adjustment according to the different data sets.

For an example, if we lower the learning rate to 0.0005 after some epochs and to 0.00005 after some another epochs, the convergence of the experimental results will be more significant.

V. CONCLUSIONS AND FUTURE DIRECTIONS

In this paper, we proposed a CNN-based methods for text classification. Our model employs a convolutional neural network (CNN) and a highway network over characters, whose output is given to a long short-term memory(LSTM) recurrent neural network model. Through experiments, it can be seen that our methods are able to achieve satisfactory classification performance with relatively small amount of training time. We can draw a conclusion that our proposed model outperforms the standard CNN and traditional models. In addition, we found the difference of classification between our baseline models has much effect on the performance of our methods.

As for the future work, we notice that in our neural networks, the multi-layer LSTM still has great room for improvement. So to conclude this work we propose here two lines for future work that we think might be interesting to check. First, since the RNNs have fixed size of memory, their memory cannot store all the information about the words it have seen before in the sentence, and thus the useful long-term information may be ignored when predicting the next words. The improved model such as attention-based memory selection recurrent network (AMSRN) was proposed so that it can review the information stored in the memory at each previous time step and select the relevant information to help generate the outputs in [31].

Second, in the spirit of substituting the LSTM-style model, the Facebook AI Research presented a convolutional approach to language modeling. The paper introduce a novel gating mechanism that eases gradient propagation and which performs better than the LSTM-style gating of despite being simpler. To our knowledge,this is the first time a non-recurrent approach outperforms strong recurrent models on these tasks in [32].So our next work is try some model basing the gated convolutional networks to replace the RNN in our model.

VI. ACKNOWLEDGMENTS

This work was supported by the State's Key Project of Research and Development Plan of China (No.2016YFC0600900),the National Natural Science Foundation of China (No.61572505, 61403394, 61402483), the Industry University Research Project in Jiangsu Province (No. BY2015023-05) and the Six Talent Peaks Project in Jiangsu Province(No. 2015-DZXX-010).

REFERENCE

[1] Kim Y. Convolutional Neural Networks for Sentence Classification[J]. Eprint Arxiv, 2014.

[2] Mikolov T, Chen K, Corrado G, et al. Efficient Estimation of Word Representations in Vector Space[J]. Computer Science, 2013.

[3] Zadrozny B. Learning character-level representations for part-of-speech tagging[C]// International Conference on International Conference on Machine Learning. JMLR.org, 2014:II-1818.

[4] Johnson R, Zhang T. Effective Use of Word Order for Text Categorization with Convolutional Neural Networks[J]. Eprint Arxiv, 2014.

[5] Zhang X, Zhao J, Lecun Y. Character-level convolutional networks for text classification[C]// International Conference on Neural Information Processing Systems. MIT Press, 2015:649-657.

[6] Srivastava R K, Greff K, Schmidhuber J. Training very deep networks[J]. Computer Science, 2015.

[7] AMandelbaum A, Shalev A. Word Embeddings and Their Use In Sentence Classification Tasks[J]. 2016.

[8] Mikolov T, Sutskever I, Chen K, et al. Distributed Representations of Words and Phrases and their Compositionality[J]. Advances in Neural Information Processing Systems, 2013, 26:3111-3119.

[9] Collobert R, Weston J, Karlen M, et al. Natural Language Processing (Almost) from Scratch[J]. Journal of Machine Learning Research, 2011, 12(1):2493-2537.

[10] Kalchbrenner N, Grefenstette E, Blunsom P. A Convolutional Neural Network for Modelling Sentences[J]. Eprint Arxiv, 2014, 1.

[11] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," in ACL,Baltimore, Maryland, June 2014, pp. 655–665.

[12] Goldberg Y. A Primer on Neural Network Models for Natural Language Processing[J]. Computer Science, 2015.

[13] Yin W, Schütze H. Multichannel Variable-Size Convolution for Sentence Classification[J]. 2016:204-214.

[14] Zhang Y, Wallace B. A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification[J]. Computer Science, 2015.

[15] Poria S, Cambria E, Gelbukh A. Deep Convolutional Neural Network Textual Features and Multiple Kernel Learning for Utterance-Level Multimodal Sentiment Analysis[C]// Conference on Empirical Methods in Natural Language Processing. 2015.

[16] Hinton G E, Srivastava N, Krizhevsky A, et al. Improving neural networks by preventing co-adaptation of feature detectors[J]. Computer Science, 2012, 3(4):págs. 212-223.

[17] Pang B, Lee L. Seeing stars: exploiting class relationships for sentiment categorization with respect to rating scales[C]// Meeting on Association for Computational Linguistics. Association for Computational Linguistics, 2005:115-124.

[18] Socher R, Perelygin A, Wu J Y, et al. Recursive deep models for semantic compositionality over a sentiment treebank[J]. 2013.

[19] Hu M, Liu B. Mining and summarizing customer reviews[C]// Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, Usa, August. DBLP, 2004:168-177.

[20] Bo P, Lee L. A sentimental education: sentiment analysis using subjectivity summarization based on minimum cuts[C]// Meeting on Association for Computational Linguistics. Association for Computational Linguistics, 2004:271.

[21] Li X, Roth D. Learning Question Classifiers[J]. Coling, 2002, 12(24):556--562.

[22] Wiebe J, Wilson T, Cardie C. Annotating Expressions of Opinions and Emotions in Language[J]. Language Resources & Evaluation, 2005, 39(2-3):165-210.

[23] Zeiler M D. ADADELTA: An Adaptive Learning Rate Method[J]. Computer Science, 2012.

[24] Le Q V, Mikolov T. Distributed Representations of Sentences and Documents[J]. 2014, 4:II-1188.

[25] Wang S I, Manning C D. Fast dropout training[C]// International Conference on International Conference on Machine Learning. JMLR.org, 2013:II-118.

[26] Mendes A C, Wichert A. From symbolic to sub-symbolic information in question classification[J]. Artificial Intelligence Review, 2011, 35(2):137-154.

[27] Zhang X, Zhao J, Lecun Y. Character-level convolutional networks for text classification[C]// International Conference on Neural Information Processing Systems. MIT Press, 2015:649-657.

[28] Zhang X, Lecun Y. Text Understanding from Scratch[J]. Computer Science, 2015.

[29] Kim Y, Jernite Y, Sontag D, et al. Character-Aware Neural Language Models[J]. Computer Science, 2015.

[30] Yu Bengong, Zhng Linbin. Chinese short text classification based on CP-CNN[J/OL].[2017-03-31].http://www.arocmag.com/article/022018-04-021.html.

[31] Liu D R, Chuang S P, Lee H Y. Attention-based Memory Selection Recurrent Network for Language Modeling[J]. 2016.

[32] Dauphin Y N, Fan A, Auli M, et al. Language Modeling with Gated Convolutional Networks[J]. 2016.

[33] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition.Proceedings of the IEEE, 86(11):2278‑2324, November 1998.

[34] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey,P. van Kleef, S. Auer, and C. Bizer. DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia. Semantic Web Journal, 2014.

[35] G. Lev, B. Klein, and L. Wolf. In defense of word embedding for generic text representation. In C. Bie-mann, S. Handschuh, A. Freitas, F. Meziane, and E. Mtais, editors, Natural Language Processing and Information Systems, volume 9103 of Lecture Notes in Computer Science, pages 35‑50. Springer International Publishing, 2015.

[36] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li. Rcv1: A new benchmark collection for text categorization research. The Journal of Machine Learning Research, 5:361‑397, 2004.