Academic News Text Classification Model Based on Attention Mechanism and RCNN

Ronghua Lin, Chengzhou Fu, Chengjie Mao(✉), Jingmin Wei, and Jianguo Li

South China Normal University, Guangzhou, Guangdong, China
{rhlin, fucz, weijingmin, jianguoli}@m.scnu.edu.cn, maochj@qq.com

Abstract. With the expeditious development of Internet technology and academic social media, massive academic news generated by academic social media have provided rich information for scholars to communicate and learn about the latest academic trends. How to effectively classify academic news data and obtain valuable information have become one of the important research directions of information science. Traditional classification methods have the problems of high dimensions, high sparseness and weak feature expression ability, etc. Deep neural network models such as CNN and RNN are also often affected by their own parameters. In this paper we present a deep neural network model based on attention mechanism and RCNN (ARCNN). We capture the context of each word and generate the word vectors with deep bidirectional LSTM layers after preprocessing. Then we use attention mechanism to calculate the attention probability distribution of news titles and contents, effectively highlighting key information. In our experiments, we use news data of academic social network SCHOLAT and Fudan University document classification set to evaluate our model and achieve better results than other widely used text classification algorithms.

Keywords: Text classification · Deep learning · Natural language processing Attention mechanism

1 Introduction

Text classification is widely used in Natural Language Processing (NLP) tasks. At present, text classification has been used in many scenarios, such as news portal websites [2], spam filtering [3] and so on. Furthermore, academic social media networks such as ResearchGate and SCHOLAT[1] have gradually emerged. Academic exchange activities have become more frequent, generating a large amount of academic news texts. It is important that how to effectively classify and manage these news data.

Traditional machine learning algorithms, including Support Vector Machine (SVM) [4], Logistic Regression (LR) [5] and so on, have already achieved satisfying results in text classification. However, most of traditional text classification works still require feature engineering, relying on features of human design, such as dictionary [6],

[1] http://www.scholat.com.

Y. Sun et al. (Eds.): ChineseCSCW 2018, CCIS 917, pp. 507–516, 2019.
https://doi.org/10.1007/978-981-13-3044-5_38

knowledge base and so on. In addition, most of these features may have high dimensions and be sparse.

Compared with traditional machine learning methods, using deep learning to solve large-scale text classification tasks can automatically extract features without human intervention such as feature engineering. Some deep neural network models like Convolutional Neural Network (CNN) and Recurrent Convolutional Neural Network (RCNN) can achieve better results. However, there are also some problems with CNN and RCNN. For example, the training time will be much longer and there are more hyper parameters that need to be adjusted.

In this paper, we present a text classification model ARCNN based on attention mechanism and RCNN, which can effectively extract texts features and classify the academic news data. In our works, we mainly study how to classify Chinese texts. Firstly, we use word vectors trained by the large corpus to represent the words, reducing the high dimensionality and sparsity of the data. Secondly, according to the characteristics of academic social news texts, we designed a deep neural network based on attention mechanism and RCNN for feature selection. Finally, we experiment with the academic news dataset of SCHOLAT and Fudan University document classification set. The results show that our model ARCNN is superior to other widely used text classification algorithms.

The remainder of this paper is organized as follows. Section 2 covers related work and Sect. 3 we introduce the details of our model ARCNN. Section 4 contains our experiments and results analysis. In Sect. 5 we summarize our works.

2 Related Work

2.1 Text Representation

The goal of text representation is to represent the texts or words that can be identified by a computer, using the formalization of language or mathematical descriptions [1]. The text representation methods that currently commonly used mainly include one-hot text representation and distributed text representation.

One-hot text representation is the most intuitive and commonly used method in NLP tasks [7]. The characteristics of one-hot representation is that each row in feature matrices has one and only one element is 1, and the other elements are 0. However, the biggest drawback is that the data is sparse.

Distributed text representation method will map words into a dense vectors space which is also with low dimension according to the relationship between current word and the context. In 2013, Google released the open source toolkit Word2Vec [8], which made word vectors widely used. Word2Vec is an unsupervised pre-training method, using low-dimensional vectors to represent words. It will put the similar words in close positions and map words into fixed-length short vectors. In this paper, we use the words vectors trained by Word2Vec as input to our model ARCNN.

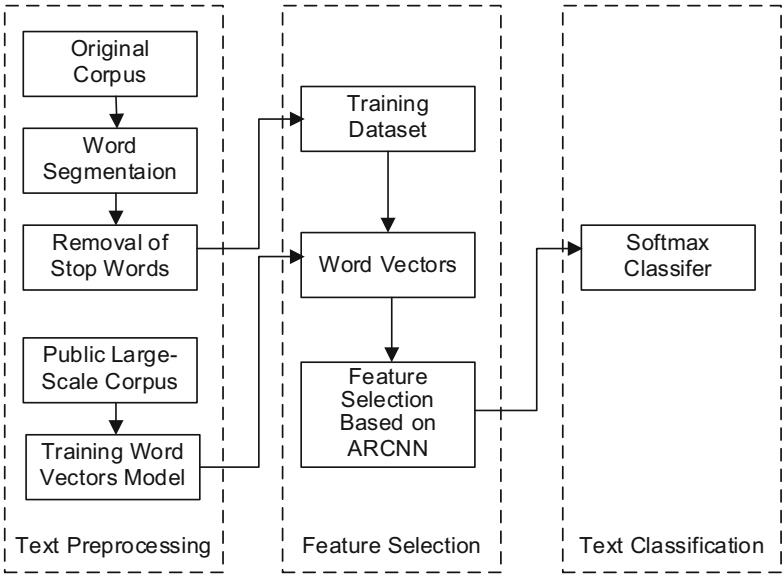

Fig. 1. ARCNN model includes three phases, (1) text preprocessing, (2) feature selection, (3) text classification.

2.2 Traditional Text Classification

Traditional text classification methods may require feature engineering before training, including text preprocessing, feature selection, and text representation. In this phase, a lot of human costs is required.

An appropriate classifier is need to be chosen after the feature engineering phase. Traditional text classification models include LR, SVM, et al. However, the learning ability of these models may be limited. They require a better feature engineering phase to analyze the effective features or feature combinations in advance, thereby indirectly enhancing the learning ability.

2.3 Text Classification Based on Deep Learning

The text classification models based on deep learning like CNN, RCNN and so on, has the following advantages [9]. Firstly, it is not required of artificial feature selection. Secondly, deep learning can utilize the features of word order well and obtain better results. Thirdly, the accuracy will be improved as the increase of training samples and the network depth.

In 2014, Kim [10] firstly applied CNN [11] to text classification. CNN model can extract deeper information from the texts. However, it is hard to set the convolutional kernel size.

RCNN model [12] obtains the context representation of each word by using the forward and backward recurrent neural networks. It can effectively extract the text features of the entire texts.

Although CNN and RCNN are effective in text classification tasks, there are still deficiencies that are not intuitive enough and poorly interpretable, especially when applied to bad situations or scenarios. Attention mechanism [13] is able to calculate the attention probability distribution for the current output, thereby highlighting the key information of the texts. Bahdanau et al. [14] used attention mechanism in translation tasks and achieved satisfying results.

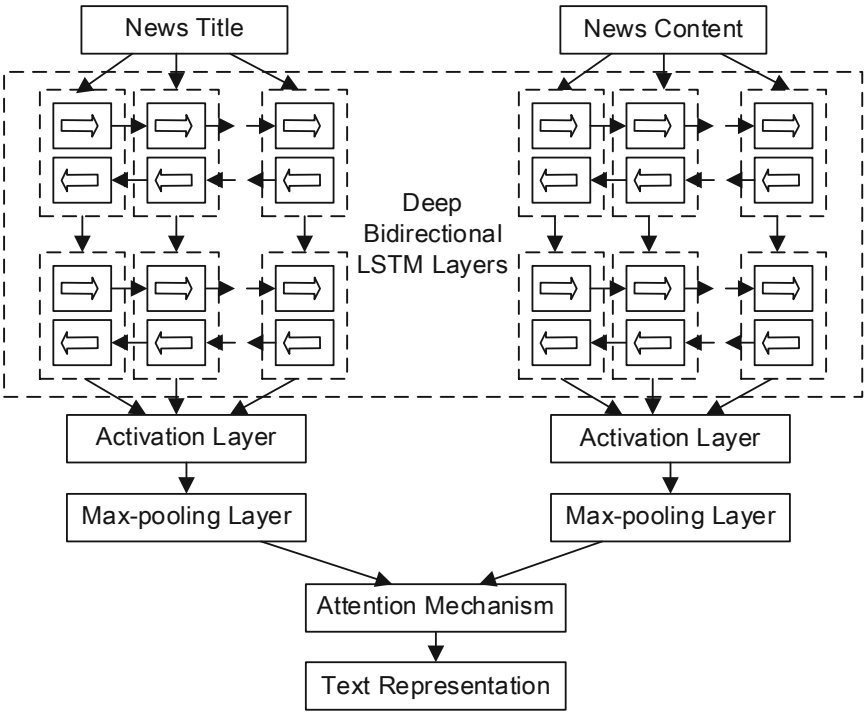

Fig. 2. Feature selection model based on ARCNN.

3 Methodology

We will then introduce the details of our model ARCNN in this section. It mainly focuses on three phases, including text preprocessing, feature selection and text classification. The pipeline is shown as Fig. 1.

3.1 Text Preprocessing

Word segmentation is an indispensable part of the Chinese text preprocessing process, which will directly determine the effect of feature selection and classification. In our

works, we use the Chinese words segmentation toolkit jieba to segment Chinese texts for further processing.

We then remove stop words for texts. Stop words refer to a large number of words in a document that are meaningless for classification, such as pronouns, conjunctions, prepositions, etc.

Since the length of the texts may affect the final results of text classification, we need to normalize the length of the texts before experiment. In this paper, we determine the length of texts by double the mean value for higher samples coverage.

When using neural networks for text classification, we need to convert the text into word vectors. However, the corpus in this paper is not large enough to train word vectors. We use 2017 Wikipedia Chinese Dataset[2] as a corpus, training by Word2Vec to generate the word vectors WV (the dimension is set to 50).

3.2 Feature Selection

We design a feature selection model based on ARCNN, taking the news titles and contents after preprocessing as input according to the composition of academic social news (see Fig. 2).

Input Layer

After text preprocessing phase, we use $\{t_1, t_2,\ldots, t_n\}$ and $\{c_1, c_2,\ldots, c_m\}$ to represent news title (T) and content (C) respectively, where n is the number of words of title and m is the number of words of content. Each word vectors of titles v_{title} and contents $v_{content}$ in T and C are respectively constructed by searching in the word vectors WV trained by Word2Vec in advance. Those that do not appear in WV are represented by random initialization. v_{title} and $v_{content}$ are as shown in Eqs. 1 and 2, where $\mathbb{R}^{V*dim}$ is the word vectors WV and V is the size of WV.

$$v_{title} = (v_{t1}, v_{t2}, \ldots, v_{tn}), v_{ti} \in \mathbb{R}^{V*dim} \tag{1}$$

$$v_{content} = (v_{c1}, v_{c2}, \ldots, v_{cm}), v_{ci} \in \mathbb{R}^{V*dim} \tag{2}$$

Deep Bidirectional LSTM Layers

The output of input layer will be feed into a deep bidirectional LSTM layers to capture the context of each word. Taking the news titles as example, we define $t_l(w_i)$ as the left context of the word w_i and $t_r(w_i)$ as the right context. $t_l(w_i)$ and $t_r(w_i)$ are calculated as shown in Eqs. 3 and 4,

$$t_l(w_i) = f\left(W^{(l)}t_l(w_{i-1}) + W^{(sl)}\mathrm{v}(w_{i-1})\right) \tag{3}$$

$$t_r(w_i) = f\left(W^{(r)}t_r(w_{i+1}) + W^{(sr)}\mathrm{v}(w_{i+1})\right) \tag{4}$$

[2] https://dumps.wikimedia.org/.

where $t_l(w_i)$ and $t_r(w_i)$ are word vectors with $|t|$ dimension. $t_l(w_{i-1})$ is the left context of the previous word w_{i-1} of the word w_i while $\mathrm{v}(w_{i-1})$ are the word vectors of word w_{i-1}. The left context of the first word in any document is $t_l(w_1)$. $W^{(l)}$ is a matrix that converts the hidden layer to the next hidden layer. $W^{(sl)}$ is a matrix that combines the semantic of the current word with the left context of next word. Function f is a nonlinear activation function. Equation 4 is similar to Eq. 3, where $t_r(w_{i+1})$ is the right context of the next word w_{i+1} of the word w_i, and the right context of the last word in any document is $t_r(w_n)$.

Activation Layer
We then use the activation function Rectified Linear Unit (ReLU) [15] to strengthen the learning ability of our model. ReLU function can reduce the amount of calculation and greatly shorten the learning period. Furthermore, it will make the output of a part of neurons as zero, which can effectively avoid overfitting [15].

Max-pooling Layer
After then, we further extract text features by using a max-pooling layer. We only take the feature with the largest score as the reserved of the max-pooling layer, while other are discarded.

Attention Mechanism
In our model, greater weights will be given to the more important information of the documents. We use attention mechanism [13] to re-weight the output of max-pooling layer. Firstly we connect the news titles y_t and contents y_c of each news document directly and get h, as shown in Eq. 5.

$$\mathrm{h}_i = [y_t; y_c] \tag{5}$$

We then use a tanh function to calculate u_i which is the hidden layer representation of h_i (as shown in Eq. 6), where W_w and b_w are the weights and bias that our model learned.

$$u_i = \tanh(W_w \mathrm{h}_i + b_w) \tag{6}$$

To further identify the importance of each word, we use softmax function to calculate the normalized attention weight matrix α_i, representing the weight of the i^{th} word (as shown in Eq. 7), where u_w is the vectors learned during the training network.

$$\alpha_i = \frac{\exp\left(u_i^T u_w\right)}{\sum_t \exp(u_i^T u_w)} \tag{7}$$

We use α_i as weights of each word, and then weight and sum them to obtain a text representation, as shown in Eq. 8.

$$y^{(2)} = \sum_{t=1}^{m} \alpha_i \mathrm{h}_i \tag{8}$$

3.3 Text Classifier

In this paper, we use softmax [16] classifier to finish the final classification. The softmax function can convert the output of multiple neurons into probability between 0 and 1 for each categorie of academic news. For softmax classifier, the absolute value of the classification result characterizes the probability of belonging to that category.

We use Algorithm 1 as follow to summarize our model ARCNN.

Algorithm 1 ARCNN's algorithm

Input: *dim* (dimension of vector)
Output: *label* (classification result)

$\mathbb{R}^{V*dim}$ (word vectors matrix)
V (size of word vectors matrix)
$Length_{title}$ (length of news titles)
$Length_{content}$ (length of news contents)
$v_{title} = (v_{t1}, v_{t2}, \ldots, v_{tn}), v_{ti} \in \mathbb{R}^{V*dim}$ (word vectors of news titles)
$v_{content} = (v_{c1}, v_{c2}, \ldots, v_{cm}), v_{ci} \in \mathbb{R}^{V*dim}$ (word vectors of news contents)
for i **to** n **do**
    $output_{title} = \text{DBi-LSTM}(v_{ti})$
end for
for i **to** m **do**
    $output_{content} = \text{DBi-LSTM}(v_{ci})$
end for
$output_{title} = activation\ (output_{title})$
$output_{title} = maxpooling\ (output_{title})$
$output_{content} = activation\ (output_{content})$
$output_{content} = maxpooling\ (output_{content})$
attention = $[output_{title};\ output_{content}]$
label = softmax(attention)
return label

4 Experiments and Results

We use scikit-learn, Tensorflow, and Keras to implement our model ARCNN. We use the news dataset of SCHOLAT and Fudan University document classification set[3] to experiment. We initialize the training epochs as 100, and early stop until the cross entropy begin to converge. In addition, we implement and compare our model with RCNN, CNN, fastText (a Facebook open source toolkit for text classification), SVM, LR algorithms.

[3] www.datatang.com/data/44139 and 43543.

4.1 Datasets

In our works, we evaluate our model with two different datasets. First is the academic social news dataset of SCHOLAT, with total 8627 documents, each of which includes titles and contents. We manually label them as 6 categories, including (1) academic news, (2) admissions, (3) recruitments, (4) call for papers, (5) school notices, and (6) others. We take a fixed length of 50 words for news titles and 651 words for contents.

Second is Fudan University document classification set, with total 17481 documents and 9 categories. We take a fixed length of 137 words for news titles and 3981 words for contents.

4.2 Results and Analysis

We divide each dataset into 10 parts randomly and evenly, 9 of which are taken as training sets and 1 as test set. We measure the performance of our model ARCNN by precision (P), recall (R) and F1 score (F1) [17].

Table 1. Experimental results for SCHOLAT dataset

		RCNN	CNN	fastText	SVM	LR	ARCNN
Academic news	P	0.849	0.862	0.840	0.845	0.809	**0.882**
	R	0.905	0.872	0.867	0.896	0.902	**0.902**
	F1	0.876	0.867	0.854	0.870	0.853	**0.892**
Admissions	P	0.923	0.915	0.913	0.862	0.824	**0.932**
	R	0.774	0.761	0.778	0.806	0.452	**0.780**
	F1	0.842	0.831	0.840	0.833	0.583	**0.849**
Recruitments	P	0.810	0.795	0.773	0.773	0.800	**0.838**
	R	0.773	0.761	0.708	0.773	0.545	**0.781**
	F1	0.791	0.777	0.739	0.773	0.689	**0.809**
Call for papers	P	0.825	0.813	0.837	0.857	0.963	**0.850**
	R	0.868	0.860	0.857	0.790	0.684	**0.882**
	F1	0.846	0.836	0.847	0.822	0.800	**0.866**
School Notices	P	0.818	0.808	0.652	0.804	0.844	**0.837**
	R	0.804	0.799	0.754	0.732	0.679	**0.821**
	F1	0.811	0.803	0.699	0.766	0.753	**0.829**
Others	P	0.912	0.902	0.872	0.891	0.862	**0.932**
	R	0.867	0.851	0.836	0.865	0.867	**0.881**
	F1	0.889	0.876	0.854	0.878	0.864	**0.906**

From the results in Table 1 and Fig. 3, the classification results of neural network classification algorithms (RCNN, CNN, ARCNN) are significantly better than traditional classification algorithms (SVM, LR), since they can effectively extract text features, and therefore enhance the accuracy of classification.

Compared with RCNN and CNN, the proposed model ARCNN have improved F1 score in all categories, showing that attention mechanism can better highlight the key information in the documents and optimize the word vectors as features. Our model is also better than fastText from the results in Table 3 and Fig. 3.

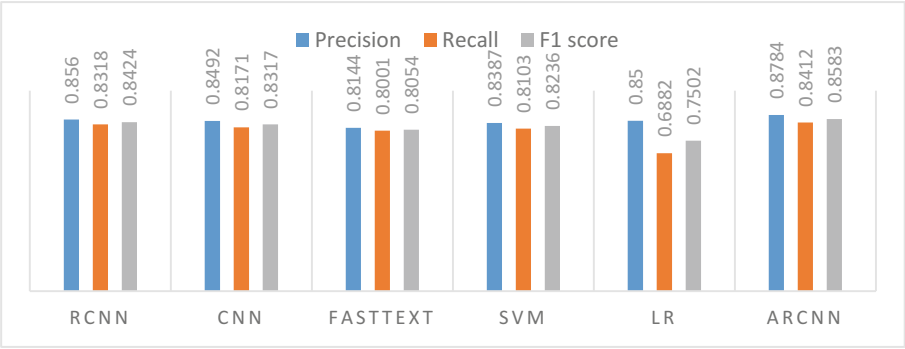

Fig. 3. The average values of precision, recall and F1 score of SCHOLAT dataset.

We also evaluate our model on Fudan University document classification set. As shown in Table 2, our model ARCNN performs better than RCNN, CNN, fastText, SVM and LR algorithms.

Table 2. Experimental results for Fudan University set

	RCNN	CNN	fastText	SVM	LR	ARCNN
P	0.944	0.937	0.931	0.935	0.929	**0.959**
R	0.946	0.937	0.924	0.927	0.907	**0.940**
F1	0.945	0.937	0.927	0.931	0.916	**0.949**

5 Conclusion

In this paper, we present a text classification model ARCNN based on attention mechanism and RCNN. We use word vectors trained by Word2Vec toolkit as text representation to avoid feature engineering phase. In the feature selection phase, we take news titles and contents as input respectively, using deep bidirectional LSTM layers and attention mechanism to further extract text features and highlight the key information in the documents. We use the academic news dataset of SCHOLAT and Fudan University document classification set to evaluate our model. The results demonstrate that our model has better performance than other text classification algorithms such as RCNN, CNN, fastText, SVM, and LR.

Acknowledgements. Our works were supported by the National Natural Science Foundation of China (No. 61772211), Science and Technology project of Guangdong (No. 2017A040405057), and "Challenge Cup" Gold Seed Cultivation Project of South China Normal University (No. 18JJKA03).

References

1. Wei, Z., Miao, D., Chauchat, J., et al.: N-grams based feature selection and text representation for Chinese text classification. Int. J. Comput. Intell. Syst. **2**(4), 365–374 (2009)
2. Zhang, H., Zhong, G.: Improving short text classification by learning vector representations of both words and hidden topics. Knowl.-Based Syst. **102**, 76–86 (2016)
3. Karthika Renuka, D., Hamsapriya, T., Raja Chakkaravarthi, M., Lakshmi Surya, P.: Spam classification based on supervised learning using machine learning techniques. ICTACT J. Commun. Technol. **2**(4), 1–7 (2011)
4. Wang, H.Y., Jian-Hui, L.I., Yang, F.L., et al.: Overview of support vector machine analysis and algorithm. Appl. Res. Comput. (2014)
5. Kurt, I., Ture, M., Kurum, A.T.: Comparing performances of logistic regression, classification and regression tree, and neural networks for predicting coronary artery disease. Expert Syst. Appl. **34**(1), 366–374 (2008)
6. Fu, C., Zeng, W., Tang, Y., Chen, L., Wei, J.: DNALS: a recommendation algorithm based on Chinese vocabulary emotion analysis of songs. Int. J. u-and e-Serv. Sci. Technol. **9**(9), 101–110 (2016)
7. Chen, X., Xu, L., Liu, Z., Sun, M., Luan, H.: Joint learning of character and word embeddings. In: International Conference on Artificial Intelligence, pp. 1236–1242. AAAI Press (2015)
8. Cui, P., Wang, X., Pei, J., Zhu, W.: A survey on network embedding. arXiv preprint arXiv:1711.08752 (2017)
9. Aggarwal, C.C., Zhai, C.X.: A survey of text classification algorithms. In: Aggarwal, C., Zhai, C.X. (eds.) Mining Text Data, pp. 163–222. Springer, Berlin (2012). https://doi.org/10.1007/978-1-4614-3223-4_6
10. Kim, Y.: Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882 (2014)
11. Wang, P., Xu, B., Xu, J., et al.: Semantic expansion using word embedding clustering and convolutional neural network for improving short text classification. Neurocomputing **174** (PB), 806–814 (2016)
12. Lai, S., Xu, L., Liu, K., Zhao, J.: Recurrent convolutional neural networks for text classification. In: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, pp. 2267–2273. AAAI Press (2015)
13. Yin, W., Schütze, H.: Attentive convolution. arXiv preprint arXiv:1710.00519 (2017)
14. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. Comput. Sci. (2014)
15. Glorot, X., Bordes, A., Bengio, Y.: Deep sparse rectifier neural networks. In: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, pp. 315–323 (2011)
16. Memisevic, R., Zach, C., Hinton, G., Pollefeys, M.: Gated softmax classification. In: Advances in Neural Information Processing Systems, pp. 1603–1611 (2010)
17. Song, F., Lin, G.: Performance evaluation metric for text classifiers. Comput. Eng. **30**(13), 107–109 (2004)