Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 208 (2022) 460–470

Procedia Computer Science

www.elsevier.com/locate/procedia

7th International Conference on Intelligent, Interactive Systems and Applications

Research on Text Classification Techniques Based on Improved TF-IDF Algorithm and LSTM Inputs

Minhui Liang[a], Tiansen Niu[b,*]

[a]Xi ' an Jiaotong University, Xi ' an 710049, China
[b]Xi ' an Jiaotong University, Xi ' an 710049, China

Abstract

Text classification is a technique that automatically classifies and labels text according to certain rules, and is widely used in sentiment analysis, intelligent recommendation systems and intelligent question and answer systems. Deep learning-based text classification methods can automatically identify and extract features in text that are useful for classification, so that it can analyse the text content directly, saving a lot of labour costs required for manual feature extraction. In this paper, the TF-IDF algorithm and the input structure of bidirectional LSTM was modified. Specifically, we rewrote the TF-IDF formula and processed the texts with a sliding window. The word2vec vector was used as the word embedding layer, and the text features were extracted using a combination of bidirectional LSTM and Text-CNN for classification prediction. Bidirectional LSTM treats texts as sequences to grasp information as a whole, while Text-CNN can extract local important features at the sentence level. As a result, good accuracy was achieved.

Keywords: Natural language processing; text classification; deep learning; hybrid neural network

1. Introduction

With the development of Internet technology, a large amount of textual information is generated daily on social networking platforms such as social media and knowledge forums, which is of high research value in areas such as intent analysis, recommendation systems, and opinion regulation [1]. Classifying text in one or more directions on the web is therefore an important technique. However, traditional text classification techniques rely on the manual design

* Corresponding author.
E-mail address: 2019867978@stu.xjtu.edu.cn

10.1016/j.procs.2022.10.064

and analysis of features, and the large volume, diverse forms and complex features of web text information make the manual design of features a problem because of high workload and low efficiency.

In recent years, deep learning models have made repeated breakthroughs in text classification [2]. Given the input text and corresponding classification labels in the training set, the models can autonomously adjust their connection weights by optimizing the losses, and construct classification functions from compound non-linear mappings [3]. Due to the good adaptive nature of deep learning models, they are able to automatically extract text features, which improves the efficiency of classification and achieves higher accuracy than traditional text classification methods.

In this paper, we used a neural network that fuses multiple hierarchies to process textual information. The word embedding layer incorporated the improved the TF-IDF algorithm and the word2vec vector, at the same time, the inputs of the LSTM have been redesigned. The experiments demonstrated that this method yielded good classification results and was a guide to research on text classification techniques in other research directions.

2. Related work

The accuracy of text classification relies heavily on the quality of the word embedding. Chenghui Huang etc. [4] make some progress in their work on measuring the similarity of short texts by combining TF-IDF values with word2vec word vectors to represent short texts.

With the development of deep learning, many classical network models of deep learning have been appropriately improved for text classification problems, greatly enhancing the effectiveness of text classification. Kalchbrenner etc. [5] conduct an exploration of using CNN for natural language processing tasks, using dynamic convolutional neural networks (DCNN) to extract text features and process information. Kim [6] proposed the Text-CNN classification model. The number of parameters and network structure were reduced and simplified to improve computational efficiency. Wang etc. [7] use CNN to extract text features and uses SVM classifier for text classification, finally good results are achieved. The CNN model performed well in short text classification tasks, but accuracy was reduced in longer texts due to excessive focus on local features.

To overcome the shortcomings of CNN in dealing with textual information problems, researchers combine LSTM with CNN to better understand the content and intent of texts by taking full account of the rich semantics contained in the context using the forgetting mechanism of LSTM. Zhang etc.[8] connect CNN layer after the LSTM layer, which effectively improves the classification accuracy; Li Yang etc. [9] base on the fusion of CNN and bidirectional LSTM networks for sentiment classification of text; Liang Shunpan etc. [10] achieve better results by fusing CNN, LSTM and attention mechanisms in a text classification task. However, the input form of LSTM can be further improved.

In this paper, the TF-IDF algorithm was improved according to the actual situation of web text classification so that it can more accurately highlight the degree of contribution of words to text classification, and the improved TF-IDF values were fused with the word2vec vector based on the CBOW method to construct the word embedding layer. In this paper, the input of the bidirectional LSTM was filtered, and the n-gram information was considered to smooth the semantics of words and highlight the contextual focus. In the construction of the deep learning model, this paper fused Text-CNN and bidirectional LSTM to extract and highlight important partial features of the text while using LSTM to take into account the contextual semantics, so as to grasp the general meaning of the text from a global perspective for better classification.

3. Experimental methods

3.1. Construction of the word embedding layer

3.1.1 Improved TF-IDF algorithm

The TF-IDF algorithm is a comprehensive assessment of the importance of a word for a text or class of texts [11]. The TF (word frequency) is the frequency of occurrence of the word in that article or class of documents, which intuitively indicates the importance of the word for that article or class of documents; the IDF (inverse document frequency) characterizes the ability of the word to differentiate against the text classification [12]. Its calculation

method is to divide the total number of documents by the number of documents containing the word. Under the premise that the total number of documents remains unchanged, the fewer the number of documents containing the word, the more special the word is, and the more suitable it can be used to identify the class of the documents containing the word [13]. The formula for the TF-IDF algorithm is shown in (1) - (3), where TF_{ij} denotes the TF value of the i-th word for the j-th document, IDF_i denotes the IDF value of the i-th word, and $TF\text{-}IDF_{ij}$ denotes the TF-IDF value of the i-th word for the j-th document. D is the total number of all documents, D_j is the j-th document, n_{ij} is the number of occurrences of the i-th word in the j-th document, and N_i is the number of occurrences of the i-th word in all documents.

$$TF_{ij} = \frac{n_{ij}}{D_j} \tag{1}$$

$$IDF_i = \lg(\frac{D}{N_i}) \tag{2}$$

$$TF - IDF_{ij} = TF_{ij} \cdot IDF_i \tag{3}$$

However, although the traditional TF-IDF algorithm can unsupervisedly find some keywords that have special significance and outstanding contribution to text classification, it also has certain shortcomings. The shortcomings are as follows.

- It does not take into account the author's diction habits. For longer, segmented texts, it is often the author's habit to begin the text with a theme and end it with a deeper idea, especially in argumentative and expository texts [14]. It is unreasonable to treat words that appear throughout the text equally, regardless of where they appear.
- The lexical nature of words is not considered. Words of different lexical nature contain different amounts of information for classification questions. Nouns contain significantly more information than verbs, verbs contain significantly more information than adjectives. A distinction should be made between words of different lexical nature in the TF-IDF formula.

In view of the above shortcomings, there have already been studies on the improvement of the TF-IDF algorithm. In this paper, the steps to improve the calculation of the TF-IDF value, taking location and lexical category into account, were as follows.

- Documents or texts were divided into words, lexical annotation was performed, and deactivated words were filtered. Words such as "of", "then" and "and" are of minimal importance to the text classification task, but due to their widespread use, the TF values can be relatively large, which not only affects the accuracy of the TF-IDF values in assessing the contribution of words to classification, but also increase the computational cost. Therefore it is important to filter these words.
- Calculate the TF-IDF value for each word according to equations (1) - (3)
- The TF-IDF values for words appearing in the first and last paragraphs of the text and for noun words were each multiplied by a higher weight to highlight the importance of these words.

The improved TF-IDF formula is shown in (4), with the i-th word appearing in the first and last segment when $f(i)$ otherwise $f(i) = 1$; $g(i)$ when the i-th word is a noun ,and $g(i) = 1$ when the i-th word has a different lexical nature. are numbers greater than zero.The remaining terms have the same meanings as in equations(1) - (3)

$$TF - IDF_{ij} = f(i) \cdot g(i) \cdot TF_{ij} \cdot IDF_i \tag{4}$$

3.1.2 Fusion of word2vec vectors with improved TF-IDF values

Word2vec is a word embedding method that maximizes the conditional probability so that the neural network can find out a word to be the predicted word when the contextual word vectors are known by the neural network [15]. The

semantic meaning of the word is fully represented [16]. In this paper, the word2vec vectors were fused with the improved TF-IDF values. The improved TF-IDF values of the words were used as weights to form a new word vector to highlight keywords that were significant for text classification and to suppress noisy words that may cause interference [17]. The largest TF-IDF value of a word for all text categories was used as the weight of its word vector when generating the word vector. The formula is shown in (5). The subscripts have the same meaning as in equations (1) - (4).

$$TF-IDF_i = \text{argmax}_j (TF-IDF_{ij}) \tag{5}$$

The new formula for calculating word vectors is shown in (6). The meaning of the subscripts is the same as in equations (1) - (5).

$$wordVector_i = TF-IDF_i \cdot word2vec_i \tag{6}$$

3.2. Improvement for LSTM inputs

The LSTM is the Long Short Term Memory Network model, as an extension of the RNN (Recurrent Neural Network) [18]. As RNN may suffer from the gradient disappearance problem when dealing with longer sequence models, LSTM introduces memory cells that consist of several kinds of gate unit, including "forget gate", "input gate" and "output gate" in each recurrent body. By learning to reasonably discard the past information stored in the state, replenishing the current input as new memory, it can generate state and output information. The calculation equations of gate units are shown in (7) - (9). Where i denotes the input gate, f denotes the forget gate and o denotes the output gate; W_i, W_f , W_o are the weights of the input gate, forget gate and output gate respectively, b_i , b_f , b_o are the biases of the input gate, forget gate and output gate respectively, h_t , x_t are the inputs and states of the loop body at time t (t-th element in the sequence) in the LSTM respectively. Since the activation function of the gate unit uses a sigmoid function, the gate unit vector components are compressed between 0 to 1. Therefore, the input or state vector is multiplied by the bitwise component of the gate vector, and the input or state is suppressed when the gate unit component converges to 0 and activated when it converges to 1.

$$i = sigmoid\left(W_i\left[h_{t-1}, x_t\right] + b_i\right) \tag{7}$$

$$f = sigmoid\left(W_i\left[h_{t-1}, x_t\right] + b_f\right) \tag{8}$$

$$o = sigmoid\left(W_i\left[h_{t-1}, x_t\right] + b_o\right) \tag{9}$$

As a result, after learning, the LSTM model can appropriately capture the information related to the semantics of the words in the context when processing a word in the text, thus better capturing the semantics of the text. The semantics of a word at a certain position in the text is not only related to the words written after the word, but also may be related to the words written before the word, so the use of one-way LSTM alone is not sufficient for mining the semantics of the text. Therefore, this paper used a bidirectional LSTM model, where the input was processed by both the forward LSTM layer and the reverse LSTM layer. The bidirectional LSTM structure is shown in Fig. 1, where X_x denotes the input, h_x denotes the state and O_x denotes the output.

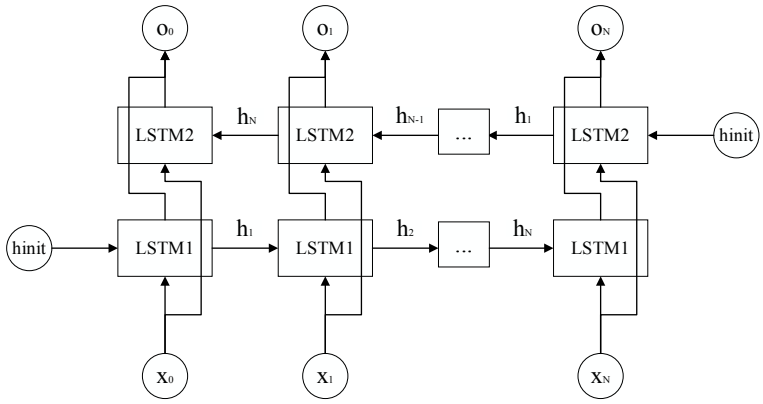

Fig. 1. structure of the two-way LSTM model.

The input of traditional LSTM models is a word vector of individual words. Although the forgetting gate has the effect of solving the gradient disappearance and can take into account the contextual semantics comprehensively, the extraction of partial features at the sentence level (length of 10 words and below) can be further enhanced. In addition, short texts such as short shopping reviews and daily blogs are characterized by more liberal editing, a high degree of colloquial expression, misuse of homophonic or morphological words, and a high number of dialectal words, raising the difficulty for text analysis.

To solve above problems, this paper improved the bidirectional LSTM input vector by adding an input window between the word embedding layer and the bidirectional LSTM layer, setting its window length to M (M is an odd number), with parameters given by standard normal distribution. The window treats the contextual word content of the target word equally, with weights decreasing evenly to the left and right sides, in line with people's general habits when reading, and normalizing the weights. When inputting, each word in the window before and after the target word (words for each side) is considered, so that the input vector is the weighted sum of the target word and the words in the window. The calculation formula is shown in (10), $Lstm\_input_i$ is the i-th cell of the LSTM, and a_j is the weight parameter of the corresponding position of the input window.

$$Lstm\_input_i = \sum_{j=i-M-\frac{1}{2}}^{i+M-\frac{1}{2}} \alpha_j \cdot word2vec_j \tag{10}$$

The input vector processed by (10) can better solve the above problem.

- Sentence-level partial features are taken into account in an integrated manner according to a normal distribution, and the LSTM model is able to grasp information more directly in the n-grams, processing the text in a way that is closer to human reading habits.
- The input window has the effect of smoothing the word vectors, making the algorithm more robust. If there is a word misuse problem at a position in the text sequence, or if the position is a word in a dialect that is not easily understood, the impact of the wrong word on LSTM learning can be mitigated by weighted summation with the adjacent word vectors; in addition, the word vectors obtained through the input window have practical significance because the addition and subtraction of word2vec vectors can represent semantic information.
- The input window can automatically perform data cleaning. For words that appear in the text but are not included in the word embedding layer, the input window can automatically infer the value of the word vector at the location of the target word based on its context.

It is worth noting that the first (M-1)/2 words and the last (M-1)/2 words of the text did not pass through the input window, so the word vector of these words was used as the LSTM input directly.

3.3. Application of Text CNN

Text CNN is a model that applies convolutional neural networks to the extraction of text features. Specifically, let the word vector dimension be V. Then, to ensure that the word vector is intact and not segmented, the size of the filters is R • V, where R can take a set of different values, and a variety of filters of different sizes are constructed to process the text matrix. The output vector of the convolution layer is then pooled to obtain the feature vector. Text CNN is able to significantly reduce network complexity and improve learning efficiency by simplifying the network structure as well as sharing parameters. The height R of the filter fully captures the textual n-gram information and is more effective in the extraction of shallow features of short texts. The structure diagram is shown in Fig. 2, where W_x is the word vector of words and N is the number of words contained in the article.

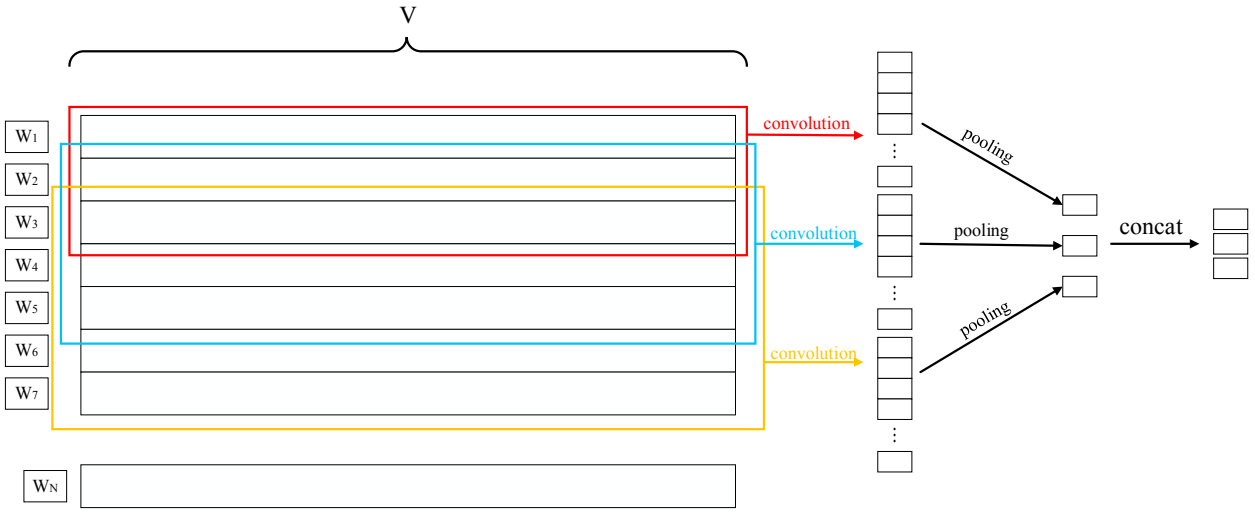

Fig. 2. Text_CNN structure diagram.

Acknowledgement and Reference heading should be left justified, bold, with the first letter capitalized but have no numbers. Text below continues as normal.

In the Text_CNN pooling layer this paper used the mean pooling method to preserve the semantic information of the n-grams and enhance the interpretability of the extracted features. If the maximum pooling method were used, the sequence information cannot be preserved and the realistic meaning of the features cannot be well interpreted. It would be also easily disturbed by noisy words and cannot well analyze the text with semantic complexity such as transitive components.

3.4. Construction of hybrid neural network layers

- The input to the hybrid neural network layer is obtained from the word embedding layer. The word embedding layer is constructed using the method described in 3.1 to convert the input text into a matrix composed of word vectors. Let the text contains N words and the dimension of word vectors be V, in which way the text is converted into a matrix of size N • V. This matrix is processed by both Text_CNN and bidirectional LSTM at the same time. For the Text_CNN, the matrix enters the convolution layer as its input; for the bidirectional LSTM, the matrix is processed by the input window as the way described in 3.2 to obtain the improved input of LSTM, a matrix with the same N• V size, which is used as the input to the bidirectional LSTM.
- The specific processing of the input matrix by the hybrid neural network. For Text_CNN, let the convolution layer takes m filters of each of the n different sizes, then their sizes are r_i • V ($0<i<n$, $0<r_i<N$); the text matrix is convolved

along the height direction, then the output vector enters the pooling layer for mean pooling to retain semantic information. The features of the connected pooling layer output form the feature vector U of the Text_CNN output. For the bidirectional LSTM, each of the Lstm_input vector which is a weighted word vector contains the n-grams, is used as input to the bidirectional LSTM, and the output vector of the LSTM cell is used as the feature vector W for the output of the two-layer LSTM. The feature vectors U and V are expanded and stitched together as Output, the feature vector extracted by the hybrid neural network.

- Classification is performed using feature vectors. Let Output be the input to the fully connected layer and use the softmax layer to output the classification results.

The hybrid neural network layer structure is shown in Fig. 3:

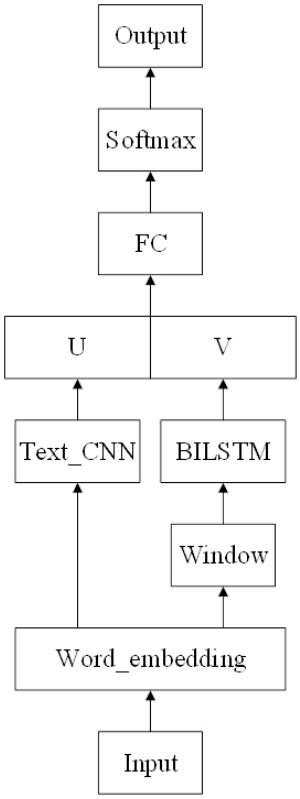

Fig. 3. hybrid neural network structure.

4. Experimental results

4.1. Experimental setup

This paper used the Sohu news dataset to train the neural network and perform validation and testing. The dataset was divided into training set, validation set and testing set in the ratio of 8:1:1. For the word embedding layer, the paper used the HanLP word segmentation function package to split words, and calculated the improved TF-IDF values of words, then obtained the 199-dimensional word2vec word vector based on the CBOW model. For the neural network layer, the paper used tensorflow2.7 and kerasAPI to build the network structure and conduct training. The other parameters of the network structure are shown in Tables 1-3.

Table 1. Text-CNN parameters.

Parameters	Value
Word vector dimension	199
Filter height	3,4,5
Filter depth	128
Number of filter channels	1
Pooling methods	Mean pooling

Table 2. Bi-LSTM parameters.

Parameters	Value
Word vector dimension	199
Number of layers	2
Hidden layer size	128
Window size	3,5,7

Table 3. Training parameters.

Parameters	Value
Activation functions	RELU
Learning rate	0.0001
Loss function	Cross-entropy
Optimization function	Adam
Number of texts	1800
Batch-size	16
Epoch	20

For the evaluation of the classification results, this paper used the accuracy A, precision P, recall R and F values to portray the goodness of the classification results, and the formulas are shown in (11) - (14).

$$A = \frac{Number\ of\ questions\ correctly\ predicted}{Total\ number\ of\ questions} \tag{11}$$

$$P = \frac{Number\ of\ correctly\ matched\ questions}{Number\ of\ correctly\ matched\ questions + Number\ of\ questions\ incorrectly\ matched\ to\ this\ category} \tag{12}$$

$$R = \frac{Number\ of\ correctly\ matched\ questions}{Number\ of\ correctly\ matched\ questions + Number\ of\ issues\ belonging\ to\ the\ category\ but\ not\ correctly\ identified} \tag{13}$$

$$F = \frac{2 \cdot A \cdot R}{A + R} \tag{14}$$

4.2. Experimental results

In this paper, text classification was performed in two separate ways, by using the Bi_LSTM model or Text_CNN model alone, and the results are shown in Table 4 and Table 5. The classification was performed for the hybrid model similarly, by removing the improved TF-IDF algorithm or the improved LSTM input with window. The results are shown in Table 6 and Table 7. Finally, the fusion models with input window lengths of 3, 5 and 7 were evaluated for classification accuracy and the results are shown in Tables 8 - 10.

Table 4. Bi-LSTM.

Value\Class	Class 1	Class 2	Class 3	Class 4	Average
A			0.871		
P	0.731	0.897	1.000	0.938	0.891
R	0.971	1.000	0.871	0.643	0.871
F	0.834	0.946	0.931	0.763	0.869

Table 5. Text-CNN.

Value\Class	Class 1	Class 2	Class 3	Class 4	Average
A			0.893		
P	0.829	0.843	1.000	0.951	0.906
R	0.971	1.000	0.771	0.829	0.893
F	0.845	0.915	0.871	0.885	0.892

Table 6. Merge-Model without TF-IDF.

Value\Class	Class 1	Class 2	Class 3	Class 4	Average
A			0.900		
P	0.878	0.854	1.000	0.897	0.907
R	0.929	1.000	0.800	0.871	0.900
F	0.903	0.921	0.889	0.884	0.899

Table 7. Merge-Model without window.

Value\Class	Class 1	Class 2	Class 3	Class 4	Average
A			0.904		
P	0.761	0.933	1.000	1.000	0.924
R	1.000	1.000	0.914	0.700	0.904
F	0.864	0.966	0.955	0.824	0.902

Table 8. Merge-Model window-size = 3.

Value\Class	Class 1	Class 2	Class 3	Class 4	Average
A			0.921		
P	0.927	0.909	1.000	0.868	0.926
R	0.914	1.000	0.829	0.943	0.922
F	0.920	0.952	0.906	0.904	0.921

Table 9. Merge-Model window-size = 5.

Value\Class	Class 1	Class 2	Class 3	Class 4	Average
A			0.942		
P	0.848	1.000	1.000	0.921	0.942
R	0.957	1.000	0.986	0.829	0.943
F	0.900	1.000	0.993	0.872	0.941

Table 10. Merge-Model window-size = 7.

Value\Class	Class 1	Class 2	Class 3	Class 4	Average
A			0.871		
P	0.910	0.753	1.000	0.900	0.841
R	0.871	1.000	0.714	0.800	0.871
F	0.890	0.859	0.833	0.900	0.871

4.3. Experimental analysis

The experiments proved that the model in this paper performed better on the text classification task. From the experimental results in 4.2, it could be seen that the modle fusing improved TF-IDF algorithm with word2vec word embedding layer, outperformed the modle using word2vec word embedding layer alone for classification; the neural network model through the input window outperformed the neural network model without the input window for classification; the neural network model fusing bidirectional LSTM and Text_CNN outperformed the neural network model using bidirectional LSTM or Text_CNN alone, and a window length of 5 resulted in the best classification accuracy.

Nevertheless, there are still possibilities for further improvement of the model. The possible directions for optimization are listed in this paper as follows.

- The probability distribution of the input window weights of the bidirectional LSTM could be further improved by using the laws of word distribution and knowledge of probability statistics to obtain a new probability distribution that better fits the human habit of reading text.
- On the other hand, it is also possible to consider window weight parameters which are randomly given and have them participated in the neural network training and learning process, so that they could be continuously improved. The process of updating the input window weights using the neural network is similar to the process of updating the filter parameters, but without the use of convolutional operations. Through learning process, the improved window weight parameters have an attention effect, which could highlight keywords in the text that are useful for understanding the semantics, the words reflects the good adaptive nature of the neural network model, but at the same time, the model might be too focused on local details, so that the complexity of it might be greatly increased, making it prone to gradient disappearance.
- Fusing more fine-grained features in semantic mining [19]. Vector modelling at letter granularity and training through a neural network model could be considered in parallel to the model in this paper. Specifically, the text will be separated into letter units to form a feature vector; the letter feature vectors would passed through the bi-directional LSTM model and the Text_CNN model along with word embedding layer at the same time; the features obtained from the word input are spliced with the features obtained from the letter input to obtain the output features, which will enter the fully connected layer. This approach allows for a more detailed and comprehensive mining of text semantics, but also increases the complexity and may leads to longer training time.

5. Summary

In this paper, we combined and optimized a variety of classical models to achieve better prediction results for text classification tasks. Specifically, in the construction of the word embedding layer, the author's word usage habit, word position and lexical influence factors were taken into account in the calculation of the TF-IDF value, which was fused with the word2vec word vectors as the word embedding layer; in the construction of the neural network layer, the bidirectional LSTM and Text_CNN were applied in parallel to capture both local and global information of the text and fuse them as features. In addition, an input window improvement was proposed at the input of the LSTM, which took into account of the n-grams by setting weights and smoothing out the possible noise. The paper also proposed directions for further optimization of the model, which might be useful and inspiring for subsequent researches in related fields.

6. Reference

[1] Su Jinshu,Zhang Bofeng,Xu Xin. Advances in machine learning-based text classification techniques[J]. Journal of Software,2006(09):1848-1859.

[2] Shervin Minaee, Nal Kalchbrenner, Erik Cambria, Narjes Nikzad, Meysam Chenaghlu, and Jianfeng Gao. 2021. Deep Learning--based Text Classification: A Comprehensive Review. ACM Comput. Surv. 54, 3, Article 62 (April 2022), 40 pages. https://doi.org/10.1145/3439726

[3] Thangaraj, Muthuraman and Muthusamy Sivakami. “Text Classification Techniques: A Literature Review.” Interdisciplinary Journal of Information, Knowledge, and Management 13 (2018): 117-135.

[4] Huang Chenghui, Yin Jian, Hou Fang. A text similarity measure combining semantic information of word items and TF-IDF method[J]. Journal of Computer Science,2011,34(05):856-864.

[5] KALCHBRENNERN,GREFENSTETTEE,BLUNSOMP. Acon volutional neural network for modelling sentences［J/OL］.arXiv Preprint,2014,2014: arXiv:1404.2188［2014-04-08］.https: / / arxiv.org /abs/1404.2188.

[6] KIM Y.Convolutional neural networks for sentence classification [EB/OL]] [2014-09-03].http://emnl2014.org /papers/pdf/EMNLP2014181.pdf.

[7] Z. Wang and Z. Qu, "Research on Web text classification algorithm based on improved CNN and SVM," 2017 IEEE 17th International Conference on Communication Technology (ICCT), 2017, pp. 1958-1961, doi: 10.1109/ICCT.2017.8359971.

[8] J. Zhang, Y. Li, J. Tian and T. Li, "LSTM-CNN Hybrid Model for Text Classification," 2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), 2018, pp. 1675-1680, doi: 10.1109/IAEAC.2018.8577620.

[9] Li Yang,Dong Hongbin. Text sentiment analysis based on CNN and BiLSTM network feature fusion[J]. Computer Applications,2018,38(11):3075-3080.

[10] Liang Shunpan, Dou Mingming, Yu Hongtao, Zheng Zhizhong. A text classification method based on hybrid neural networks[J]. Computer Engineering and Design,2022,43(02):573-579.DOI:10.16208/j.issn1000-7024.2022.02.038.
[11] Jones, Karen Sparck. "A statistical interpretation of term specificity and its application in retrieval." Journal of documentation (1972).
[12] Shi, Lin,Xu, Rui-Long. Research on deep learning model based on Word2vec and improved TF-IDF algorithm[J]. Computer and Digital Engineering,2021,49(05):966-970.
[13] Zhang Lei,Jiang Yu,Sun Li. An improved TF-IDF text clustering method[J]. Journal of Jilin University (Science Edition),2021,59(05):1199-1204.DOI:10.13413/j.cnki.jdxblxb.2020347.
[14] Jing L,He T-T. A Chinese text classification model based on improved TF-IDF and ABLCNN[J]. Computer Science,2021,48(S2):170-175+190.
[15] MIKOLOVT，CHENK，CORRADOG，et al. Efficient estimation of word representations in vector space［C］// Proceedings of International Conference on Learning Representations. Scottsdale，USA：［s. n. ］，2013：1-12.
[16] Wang Gensheng,Huang Xuejian. Convolutional neural network text classification model based on Word2vec and improved TF-IDF[J]. Small Microcomputer Systems,2019,40(05):1120-1126.
[17] Peng, J.L., Gu, Y., Zhang, Z., Geng, S.H.. Fusion of word contribution degree and Word2Vec word vector for document representation[J]. Computer Engineering,2021,47(04):62-67.DOI:10.19678/j.issn.1000-3428.0056370.
[18] Hochreiter S,Schmidhuber J.Long short-term memory [J].Neural computation,1997,9（8）:1735-1780.
[19] Wang Yan,Wang Hu Yan,Yu Ben Gong. Research on Chinese text classification based on multi-feature fusion[J]. Data Analysis and Knowledge Discovery,2021,5(10):1-14.