ISSN 2616-5767 Vol.5, Issue 1: 53-62, DOI: 10.25236/AJETS.2022.050111

Label Oriented Hierarchical Attention Neural Network for Short Text Classification

Feng Xia[1,2,a*]

[1]*Ping An Healthcare and Technology Co Ltd, Shanghai, China*
[2]*Renmin University of China, Beijing, China*
[a]*feng.xia.cs@gmail.com*

Abstract: *Text classification task is a very common task in natural language processing. The conventional text classification task model often uses word bag model or representation model, but the existing model usually deals with long text classification task, which is not suitable for short text classification. Short text features are relatively fewer and often need more sophisticated feature extraction, and the role of keywords play great importance roles in the classification of short text. In this paper, we propose a label-oriented hierarchical attention mechanism network for short text classification. The model achieves better results on public data sets of Tiao and Weibo, compared with convolutional neural network CNN, GATE control unit neural network GRU, gate control unit neural network fusion convolutional neural network GRU-CNN and translation model Transfomer. It is proved that the model has good performance. Our model has two significant advantages: (1) it is a hierarchical structure consisting of two levels of attention mechanisms, which facilitates interpretation and analysis; (2) Compared with other architectures, this model can be used to extend multi-label text classification tasks. In addition, it can be used for long-term importance analysis in many industrial scenarios.*

Keywords: *short text classification; Hierarchical Attention; term importance; Convolution network; Natural language processing; Neural Network*

1. Introduction

With the rapid development of Internet update iteration, more and more text information are produced. As one of the data processing directions of text information text classification is a very common task in natural language processing. The goal is to obtain functions that accurately map text to one or more of the pre-given tags. In the industry, text classification has many application scenarios, such as sentiment analysis, public opinion monitoring and news classification in the field of news, user intention recognition in the field of e-commerce, mail filtering and other scenarios. There are two main methods of text classification, which are based on traditional machine learning and deep learning. The traditional machine learning classification methods of text classification mainly include the process of text data preprocessing, feature extraction, text vectorization, training set modeling and prediction by using classification machine learning algorithm. For example, a word bag model is used to represent sentences, ignoring the original order of tags, and then a linear model or kernel approach [1] is used to map sentence representations to sentence labels. The disadvantage of traditional machine[2] learning algorithms for text classification is that feature extraction is highly dependent on manual work. Deep learning classification method of text classification mainly uses neural network model such as CNN to train and predict data to complete classification, which can automatically complete feature extraction through the model. In recent years, with the development of deep learning, more and more methods use randomly initialized continuous vector to represent word representation, and adjust weight distribution through back propagation algorithm.

Since this paper mainly adopts deep learning method to achieve text classification, it focuses on introducing several algorithms of text classification based on deep learning, and illustrates their advantages and disadvantages. The text classification method with neural network is mainly based on convolution neural network [3,4,5,6,7,8], recurrent neural network [9,10,11] and attention neural network [12,13]. Convolution neural network can capture local features and support parallel computing. As a memory network, recurrent neural network can effectively remember tokens information which appear in sentences. Kim [4] proposed a convolution network for text classification, but did not consider the importance difference of words. Lai [10] put the whole sentence into the recurrent neural network and

ISSN 2616-5767 Vol.5, Issue 1: 53-62, DOI: 10.25236/AJETS.2022.050111

take the final output embedding as a sentence representation. The recurrent neural network takes the output of previous state as the current input, which takes too much time for training, and is not convenient to analysis the importance of each word. Yang [12] proposed a network structure based on attention mechanism, which contains level attention mechanism and sentence level attention mechanism. However, the design of attention mechanism does not distinguish the importance of different words between different labels. For example, "trump will be on the American talent show" has two labels of "entertainment" and "politics". Under the "entertainment label", the "American talent show" would play a main role. For the "political" label, it is obvious that we should pay more attention to the word "trump".

Short text classification is a kind of text classification. Compared with long text, short text has the characteristics of fewer words, relatively weak description information, relatively clean and non-standard. Because of the disadvantages of using traditional machine learning methods to do text representation, such as high sparsity, weak feature representation ability, high cost, and the need for extra manual feature engineering, it cannot well meet the demand of short text classification. As a result, deep learning has gradually penetrated into the field of natural language processing due to its advantages of automatic feature extraction from the initial application in image. Naturally, deep learning[14,15] relies on the ability to automatically acquire feature representation and is also used to solve short text classification problems. At present, the main deep learning methods for short text classification include TextCNN, TextRNN, TextRCNN, Transfomer[16,17,18] and other neural network algorithms.

Our original intuition is to strengthen the importance of key token information. For convolution networks, the influence of each sliding window on classification is different, and is mainly determined by label tagged on texts. Therefore, we design a label oriented attention mechanism for judging the importance of different convolution windows; In this paper, we use a variety of convolution for extracting texts features. For each text feature embedding after convolution, we employ the attention mechanism to gather all the results. Our work has two main contributions: (1) this paper designs a structural attention mechanism network convergence feature, which can show the contribution of each word to the final result. (2) The short text classification model proposed in this paper can be used for multi label tasks and has very good scalability.

2. Architecture

In this chapter, we propose a novel label based attention mechanism network for text classification, which can effectively capture the feature information of text. The overall architecture of this paper is shown in the figure 1. It is composed of three parts: the first part is multi kernel convolution layer, which extracts the feature information of the text from different receptive fields. The second part is the label oriented attention mechanism layer, which can aggregate the information of the token features and generate the sentence vector from the token vectors. After that, we use attention mechanism as pooling layer to merge multiple channels sentence representation into the final sentence representation. The third part, we take each label classification as individual event and design output network to predict final label for each text.

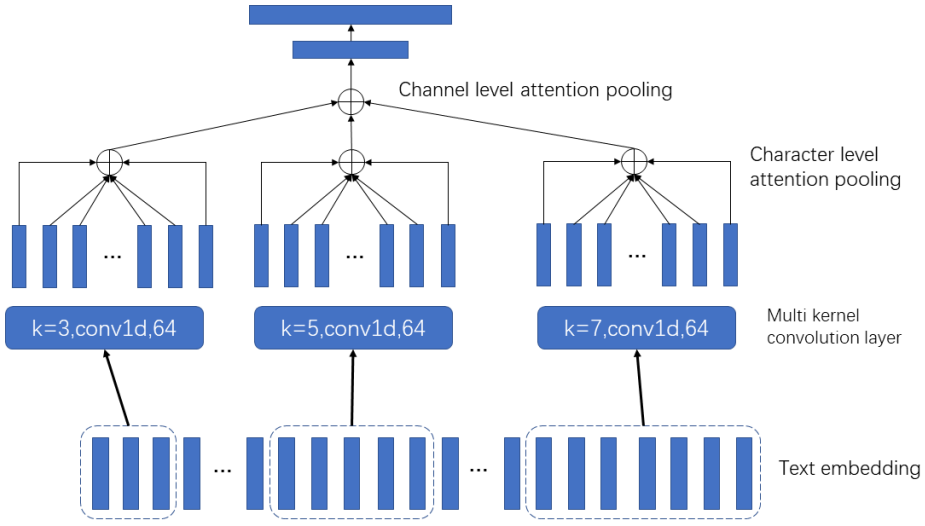

Fig.1. Label Oriented Hierarchical Attention Neural Network Framework

ISSN 2616-5767 Vol.5, Issue 1: 53-62, DOI: 10.25236/AJETS.2022.050111

2.1 Multi kernel convolution layer

As a simple and efficient feature extractor, convolutional neural network is very suitable for text classification. Different from the recurrent network, the computation of each convolution unit can be performed at the same time, which greatly reduces the time of training. We apply convolution network firstly to effectively extract the local features of samples, which is very useful in text classification tasks, because the keywords in the text often play an important role in text classification. We encode each token into one-hot representation and multiply with initialized word vector matrix E_D, $E_D \in R^{wn \times d}$, wn is the total number of words, d is the vector dimension of words. We set d as the same size of word dimension used in Bert. When given one-hot encoded text sequence $e_t = \{e_1, e_2, e_3 \dots, e_s\}$, we can get text sequence embedding $E_w = \{E_1, E_2, E_3 \dots, E_i \dots\}$ with word vector matrix E_D.

$$E_i = E_I E_D \tag{1}$$

In order to extract the features of tokens, we use 1D convolution neural network with different kernel size extract text features from different receptive fields, which is also used in [12].The calculation formula of convolution is as follows:

$$c_i = f(w \cdot E_{i:i+h-1} + b) \tag{2}$$

$E_{i:i+h-1}$ denotes the token representations under i-th convolution window with size of h. We set max sequence size to 50 and pad zero when tokens length is smaller than 50, otherwise truncate tokens when they are over max length. All text vectors under the convolution window are calculated with the convolution kernel weights and convolution kernel weights is the same size of the convolution window. Then when the window continuously moves from left to right, we get a dozen of convolutional features and concatenate them together as final token features. In this paper, we use different convolution kernel size which contains several filters to extract features at the same time. The features are denoted as $E_c, E_c = \{E_c^1, E_c^2, E_c^3 \dots E_c^n\}, E_c^i \in R^{s \times d}$ and s is the sequence length, d is the dimension of text features. The whole process is show as figure 2.

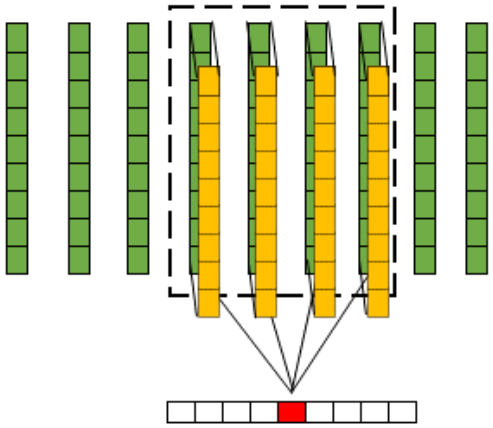

Fig.2. Text Convolution neural network

2.2 Label Oriented Hierarchical Attention

The structure of attention mechanism proposed in this paper mainly includes two parts. The first part is as a pooling layer to combine each output under the same convolution window. Each output under the convolution window is regarded as token feature in the same position of the sentence. The second part is designed to concatenate the result after the first part. In other words, we get different channels of text features after using different kernel size convolution layer, attention pooling layer and the second part is select those channel features with high degree of credibility. In order to filter tokens that are unrelated to label, we design a label oriented attention mechanism. Attention mechanism is consisted of query, key and value. After the query is given, the related information of value is returned. The correlation is determined by the matching score between query and key. In this paper, we use label as query with initialize a vector E_l , $E_l = \{E_l^1, E_l^2, E_l^3 \dots E_l^{\mathrm{m}}\}$, m is the number of labels and the vector of each label is $E_l^{\mathrm{k}} \in R^{1 \times d}$. After Multi kernel convolution layer, we get the feature vector $E_c = \{E_c^1, E_c^2, E_c^3 \dots E_c^n\}$ E_c^i is the i-th convolution for the input text, $E_c^i \in R^{s \times d}$ and s is length of the text and d is the vector

ISSN 2616-5767 Vol.5, Issue 1: 53-62, DOI: 10.25236/AJETS.2022.050111

dimension. E_c^i is the convolutional feature of convolution generated by sliding in sequence according to a certain step size of the sliding window, so the position information of words is retained, and each position vector corresponds to the word information of the same position. The first part of the attention mechanism proposed in this paper mainly evaluates the importance of words in the text. The key and value vectors are determined by E_c^i which are generated through the full connection layer, and the formula is as follows:

$$K_c^i = Relu\left(E_c^i W_k^i + b_k^i\right) \quad (3)$$

$$V_c^i = Relu(E_c^i W_v^i + b_v^i) \quad (4)$$

Where K_c^i、$V_c^i \in R^{s\times d}$ and calculated score $\widetilde{\pi_c^i}$ of label and K^i is formula as follows:

$$\widetilde{\pi_c^i} = K_c^i E_l^{kT} \quad (5)$$

We normalize all scores as:

$$\pi_c^i = \frac{\exp(\widetilde{\pi_c^i})}{\sum \exp(\widetilde{\pi_c^j})} \quad (6)$$

π_c^i is the convolution attention score for i-th position features with label k and can show the word related importance between words and the text labels. Then, according to the position score of the whole words, we weighted sum all the word features as the sentence features, and the vector representation of the sentence after convolution is denoted as $\widetilde{E_c^i}$, $\widetilde{E_c^i} \in R^{1\times d}$

$$\widetilde{E_c} = \sum \pi_c^i V_c^i \quad (7)$$

After the first layer of attention, we get sentence representations of all types of convolutions, $E_c = \{\widetilde{E_1}, \widetilde{E_2} \dots, \widetilde{E_l}\}$. l is the number of different kernel sizes. Similar to the first attention calculation, the second attention calculation also uses the label vector as the query vector, and the coefficients assigned to the vector after each convolution are calculated. All convolutional features are aggregated as E_o according to the attention coefficients.

$$E_o = \sum \pi_c E_1 \quad (8)$$

For the final output of the attention layer, after performing the same operation on each label, we obtain multiple fused feature vector sets denoted as $E_o = \{E_o^1, E_o^2 \dots, E_o^m\}, E_o \in R^d$.

2.3 Text classification

The memory mechanism layer calculates the correlation between the text and all labels, so it can be processed whether it is a single-label text classification task or a multi-label text classification task. For each label is a binary classification problem. Text labels contain label i, we can denote i-th element as one in labels one-hot encode table. Text labels do not contain label i, we can denote i-th element as zero in labels one-hot encode table instead. In this paper we design two methods for predict labels. The first method transforms all text feature vector into one dimension and then take nonlinear function sigmoid which ensure that the value of score for text ranges from zero to one. The calculation formula is as follows:

$$P_o^j = sigmod(E_o^j W^j + b^j) \quad (9)$$

$W^j \in R^{d\times 1}$, $b^j \in R^1$, P_o^j indicating whether the text has a label. $P_o^j \in R^1$

At this time, the loss function loss is:

$$Loss_{base} = -\sum_j (y^{t_j} * \log\left(P_o^j\right) + (1 - y^{t_j}) * \log\left(1 - P_o^j\right)) \quad (10)$$

The second way regards whether to have the label as a binary classification problem, and the output vector has two dimensions.

$$P_o^j = softmax\left(E_o^j W^j + b^j\right) \quad (11)$$

$W^j \in R^{d\times 2}$, $b^2 \in R^2$, $P_o^j \in R^2$. At this point the loss function becomes:

$$Loss_{base} = -\sum_j \sum (y^{t_j} * log\left(\dot{P_o^j}\right) + (1 - y^{t_j}) * log\left(1 - \dot{P_o^j}\right)) \quad (12)$$

ISSN 2616-5767 Vol.5, Issue 1: 53-62, DOI: 10.25236/AJETS.2022.050111

3. Experiments

In this section, we conduct experiments on two data sets of text classification, and use corei7-10700 processor, 64GB running memory, Nvidia RTX3060 GPU, windows10 operating system, and Python3 to write the code. We use Google open deep learning framework Tensor Flow and Jetbrain Pycharm as IDE.

3.1 Data sets

We conducted experiments on two publicly available data sets. Our experimental data sets are toutiao news data and weibo sentiment analysis books respectively. The toutiao news data includes 15 categories, including livelihood stories, culture, entertainment, sports, finance and economics, real estate, automobiles, education, science and technology, military, tourism, international, stocks, agriculture, rural areas, and games. The toutiao news dataset contained 382,675 pieces of data, with an average of 25,511 pieces of data per category tag, 41,543 pieces of data for the most categories and 340 pieces of data for the least categories. There are four categories in the weibo emotion analysis data set, which are joy, disgust, anger and depression. The weibo data set contains a total of 361,744 pieces of data, with an average of 90,436 pieces of data per category label. The category label with the largest number has 199,496 pieces of data, and the category label with the lowest number has 55,267 pieces. toutiao data sets download address is: https://github.com/aceimnorstuvwxz/toutiao-text-classfication-dataset . Weibo sentiment analysis data download address is: https://github.com/SophonPlus/ChineseNlpCorpus.

The quantitative statistics of each category of the two data sets are shown in the table:

Table 1. Toutiao news dataset statistics

entertainment	39396	automobiles	35785	culture	28031
science and technology	41543	sports	37567	games	29300
stocks	27084	education	27058	international	26909
military	24984	tourism	21416	agriculture	340
real estate	17667	rural areas	6273	livelihood stories	19322

Table 2. Weibo dataset statistics

joy	199496	anger	51714
disgust	55267	depression	55267

3.2 Baselines

In order to more scientifically clarify the effectiveness of the neural network model of the label-oriented attention mechanism proposed in this paper, we take the following four models as our baseline models. We will compare all baseline models on both datasets:

1) TextCNN[4]: The model can be divided into four network layers, namely, word embedding layer, convolution layer, pooling layer, full connection layer and Softmax layer. Embedded in the word layer to change the initial vector, each word in the convolution neural network is utilized to extract the text, the characteristics of the use of pooling layer characteristics on characteristics after the convolution filtering, retain the most useful feature information, and finally the full connection layer and softmax layer will map features to each label, assign a score to each label and predict the final results.

2) TextGRU[11]: GRU is a variant of LSTM and also a temporal neural network model. Text GRU is an application model of GRU in the field of Text classification. The model structure includes input layer, bi-directional GRU layer, full connection layer and output layer. Input layer converts text to feature representation, bi-directional GRU layer extracts useful feature information from text representation, full connection layer maps each feature information to multiple labels, and finally the results are normalized through softmax layer.

3) TextGRU+CNN[13]: This model is mainly the fusion of textCNN model and TextGRU model, realizing the complementary purpose of CNN and GRU. The model results include input layer, bidirectional GRU layer, CNN layer, full connection layer and output layer. The input layer provides the vector representation of text, the bidirectional GRU layer extracts the vector representation of text, and the CNN layer processes the features extracted by GRU to extract more useful information. The full

ISSN 2616-5767 Vol.5, Issue 1: 53-62, DOI: 10.25236/AJETS.2022.050111

connection layer maps each feature information to multiple labels, and finally normalizes the results through softmax layer.

4) Transfomer[16]: The transfomer model is entirely based on sub-attention mechanism, and the model results are mainly divided into two parts: encoding and decoding. It uses the self-attention mechanism to calculate the weight dependence between each word, which makes up for the disadvantage of RNN model that cannot be calculated in parallel, and has stronger interpretability.

3.3 Experimental Results and Analysis

We did not preprocess any data, because the short text data itself is limited in length, any character in the short text may contain useful information, we directly input the original short text data into the model. In the data analysis of 3.1, we can see that the data of different categories of labels have the problem of unbalanced samples. In order to solve the problem of unbalanced samples, we carried out the operation of negative sampling for those categories of data with insufficient data volume, and ensured the same number of samples with different labels. We then shuffled all the data and randomly fed it into the model. In the training process, we used dropout technology. While training the model, we set dropout to 0.5, randomly discarding 50% of the neurons. When the model predicts, we keep all the neurons.

On the toutiao news dataset, our experimental parameters are set as follows: When using textCNN model, we adopted l2 regular term coefficient of 0.00005 and learning rate of 1e-3, and batch size of 128. When using textGRU model, we adopt the coefficient of 0.0005 l2 regular term and the learning rate of 1e-3, and use the batch size of 256. When using textGRU-CNN model, we adopt the coefficient of 0.0005 l2 regular term and the learning rate of 1e-4, and use the batch of 128. Using the Transformer model, we used a 0.0005 L2 regular term coefficient with a 1e-7 learning rate and a batch of 256. We set the number of Transformer layers to 2.

On the weibo sentiment analysis data set, our experimental parameters are set as follows: When using textCNN model, we adopted l2 regular term coefficient of 0.00001 and learning rate of 1e-4, and batch size of 128. When using textGRU model, we adopt the coefficient of 0.0001 L2 regular term and the learning rate of 1e-5, and use batch size of 256. When using textGRU-CNN model, we adopt the coefficient of 0.00001 l2 regular term and the learning rate of 1e-5, and use the batch size of 128. Using the Transformer model, we used a 0.00001 L2 regular-term coefficient with a 1e-7 learning rate and a batch size of 256. We set the number of Transformer layers to 2.

Table 3. Experimental results on toutiao and weibo dataset

Dataset	Model	Precision	Recall	F1	Acc
toutiao news	CNN	0.84	0.84	0.84	0.84
	GRU	0.83	0.82	0.83	0.83
	GRU-CNN	0.84	0.83	0.84	0.85
	Transformer	0.81	0.80	0.81	0.82
	Our model	**0.85**	**0.87**	**0.86**	**0.86**
weibo	CNN	0.45	0.47	0.46	0.47
	GRU	0.43	0.44	0.44	0.44
	GRU-CNN	0.45	0.45	0.45	0.45
	Transformer	0.41	0.42	0.42	0.43
	Our model	**0.46**	**0.47**	**0.47**	**0.48**

Our model achieved the best results on two experimental data points, with an average improvement of 2% over the other models. Among other models, textCNN has achieved the best effect, because convolutional neural network is very suitable for extracting key information, especially some short texts. Transformer models are mediocre because Transformer models use a self-attention mechanism, which requires a lot of data to learn good distribution. GRU and GRUCNN model have a very long training time, and CNN has certain advantages over GRU in short text classification task. There is a tag-based attention mechanism in our model, which makes it easier to find useful feature information when using convolutional network to extract features.

Figure 3 shows the F1 indicators of the model in this paper in all categories. As can be seen from Figure 3, F1 indexes of all categories are basically above 85%. The model is most accurate for the text prediction of the stock category because the articles about the stock topic have very obvious characteristic

ISSN 2616-5767 Vol.5, Issue 1: 53-62, DOI: 10.25236/AJETS.2022.050111

information. Our model has very poor text prediction for agricultural category labels, because there are very few texts with agricultural labels in the whole data set, and the model has not learned enough feature information for a very small number of category data, so it is in an under-fitting state.

Fig.3. Toutiao new text classification f1-score

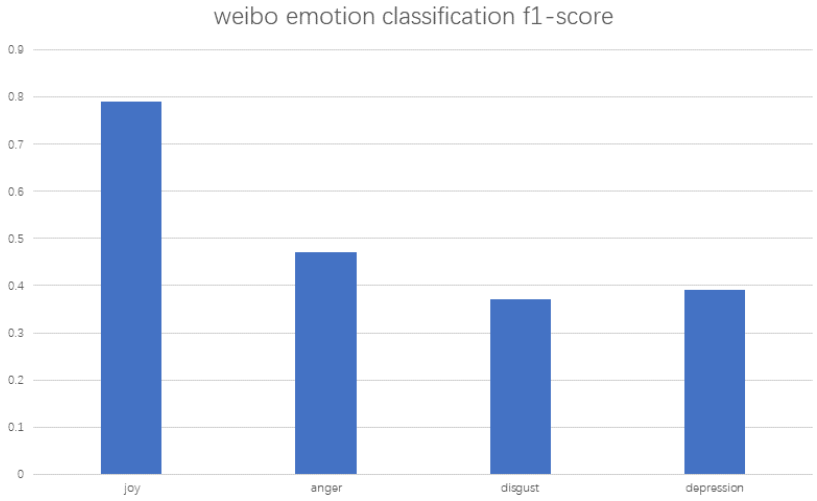

Fig.4. Weibo emotion classification f1-score

On weibo emotional analysis of the data set, all the indicators model are much lower than the headlines data set, this is because the weibo user personal data often are created, text tend to be casual, contains a large amount of complicated information, and headline data sets are often professional written text, the author of a certain professional degrees, and the text is relatively uniform. At the same time, we find that the index of the data labeled joy is very high, because the data labeled Joy is clearly differentiated from other types of data, while the labels anger, Disgust and depression are all negative emotions, and the three labels are not very differentiated. In particular, the semantics of the tag "Disgust" and the tag "depression" are so close that it is difficult for the model to distinguish them.

Table 4. Training time of each baseline model

Model	TextCNN	TextGRU	TextGRU-CNN	Transformer	Our model
toutiao news dataset	1h	6h	7h	6h	2h
weibo data	0.5h	6h	6.6h	5h	1.5h

By comparing the data in the above table, it can be seen that the training time of TextGRU, TextGRU-CNN and Transformer is very long, almost three times that of other models, among which TextGRU-CNN has the longest training time. TextCNN is the model with the shortest training time. Because its

ISSN 2616-5767 Vol.5, Issue 1: 53-62, DOI: 10.25236/AJETS.2022.050111

structure is very simple, the training time of our model is slightly longer than that of TextCNN, but our model's training convergence is also very fast.

3.4 Case Study

In order to further verify the ability of our proposed model to extract word features and explore the role of the attention mechanism in extracting features, we randomly selected some samples from the test set for analysis, and we visualized the attention mechanism. The attention coefficient of each word contains two parts, one is the coefficients assigned to different convolution windows, and the other is the coefficients assigned to different convolution kernel functions. In addition, each convolution window is performed by window translation, that is, a certain word will appear in multiple windows, so the corresponding window weights need to be added and averaged. The weight calculation formula of word w is as follows:

$$ATT_w = \sum_k w_k \frac{\sum_c w_c}{n} \qquad (13)$$

Where is the weight assigned to the label by the convolution kernel k, is the weight assigned to the convolution window c where the word appears, and n is the number of times the word appears in the convolution window $w_k w_c$.

We randomly selected 8 samples from data in the validated dataset. The attention weight distribution is calculated for each sample, and the following is the attention distribution diagram:

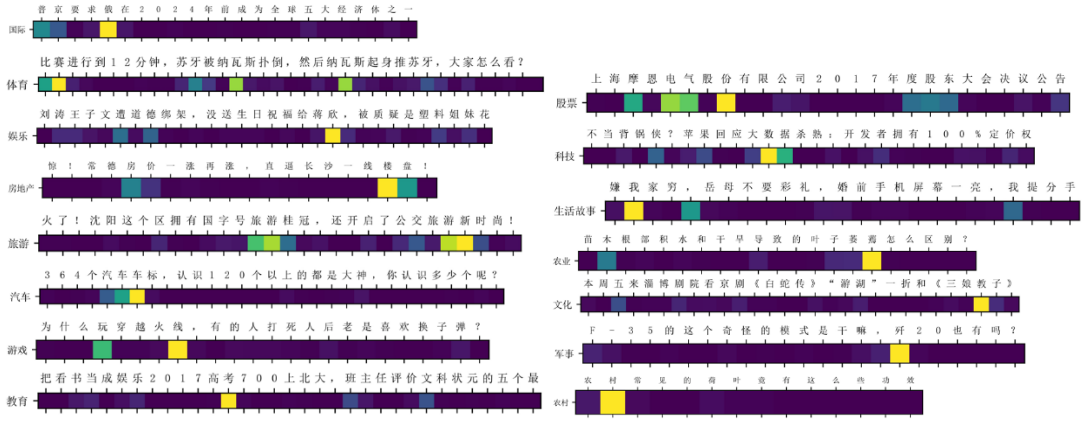

Fig.5. Visual analysis of attention weight

Figure 5 shows the contribution of each word to the text label. We randomly extract a piece of data from each label for visual analysis. As can be seen from the figure, for the rural label, the word "village" contributes most of the weight; For car labels, "car labels" contribute a lot of useful information. For tech labels, "big data" contributes a lot of useful information. For educational labels, "exams" contribute a lot of useful information. For travel labels, "travel" contributes a lot of useful information, etc. For these very characteristic labels, attention weight will appear very reasonable. At the same time, we also found some bad cases. For example, for "life story", attention weight is mainly focused on "me", which is very unreasonable. This is because the "life story" label is a very general label, a very abstract concept, containing a lot of trivial text with no obvious relevance. The model does not handle this abstract and does not clutter tag very well. In general, the attention mechanism designed in this paper captures the information of each label very well, allocates appropriate weight to each word reasonably, and filters out irrelevant information. The model proposed in this paper has very good interpretability and data processing ability.

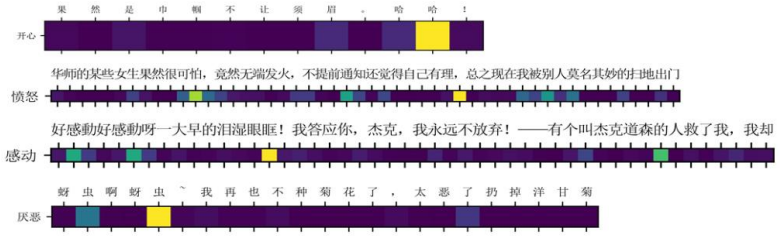

Fig.6. Visual analysis of attention weight

ISSN 2616-5767 Vol.5, Issue 1: 53-62, DOI: 10.25236/AJETS.2022.050111

In the figure above, we can see the association between each word and emotional label. The distribution of weights learned by the model is not as good as the toutiao news data set, because the texts posted by users tend to have fewer emotional features or are full of words with various emotions. In addition, many texts often contain several emotional information at the same time, and there are large deviations in the annotation of data sets.

4. Conclusion

In this paper, we propose a label-based attention mechanism network for short text classification tasks, which can effectively extract keywords in short texts. It has a two-layer attention mechanism architecture and has very strong predictability. Interpretation and the ability of text multi-label prediction. In this paper, we compare the model proposed in this paper with other models. After experimental analysis, our model achieves the best effect and verifies the correctness of the model. At the same time, we carried out visual analysis of the results, and the experiment proved that the attention mechanism designed by us could effectively extract useful information. More weight information was allocated to the key words, and less weight was allocated to those irrelevant words. In addition, this model can be used for multi-label text classification tasks, without the need to modify the model architecture. At the same time, we also noticed some bad cases, which were caused by the excessively scattered weights of the word-level vectors. Meanwhile, for the label vector analysis, we found that the distinction between label vectors was not large enough, which may also lead to a certain distraction of attention and have a certain impact on the results. In the future, we plan to introduce some additional restrictions to enhance the distinction between labels, and explore more ways of attention calculation to facilitate better convergence of the model.

References

[1] Thorsten Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. Springer.
[2] Ikonomakis, M., Sotiris Kotsiantis, and V. Tampakas. "Text classification using machine learning techniques." WSEAS transactions on computers 4.8 (2005): 966-974.
[3] Kalchbrenner, Nal, Edward Grefenstette, and Phil Blunsom. "A convolutional neural network for modelling sentences." arXiv preprint arXiv: 1404.2188 (2014).
[4] Kim, Yoon. "Convolutional Neural Networks for Sentence Classification." EMNLP (2014)
[5] Luan, Yuandong, and Shaofu Lin. "Research on text classification based on CNN and LSTM." 2019 IEEE international conference on artificial intelligence and computer applications (ICAICA). IEEE, 2019.
[6] Wang, Haitao, et al. "A short text classification method based on N-gram and CNN." Chinese Journal of Electronics 29.2 (2020): 248-254.
[7] Liu, Zhenyu, et al. "Multichannel cnn with attention for text classification." arXiv preprint arXiv: 2006.16174 (2020).
[8] Wang, Shiyao, Minlie Huang, and Zhidong Deng. "Densely connected CNN with multi-scale feature attention for text classification." IJCAI. 2018.
[9] Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." Neural computation 9.8 (1997): 1735-1780.
[10] Lai, Siwei, et al. "Recurrent convolutional neural networks for text classification." Twenty-ninth AAAI conference on artificial intelligence. 2015.
[11] Zulqarnain, Muhammad, et al. "Efficient processing of GRU based on word embedding for text classification." JOIV: International Journal on Informatics Visualization 3.4 (2019): 377-383.
[12] Yang, Zichao, et al. "Hierarchical attention networks for document classification." Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies. 2016.
[13] Yu, Shujuan, et al. "Attention-based LSTM, GRU and CNN for short text classification." Journal of Intelligent & Fuzzy Systems 39.1 (2020): 333-340.
[14] Kowsari, Kamran, et al. "Text classification algorithms: A survey." Information 10.4 (2019): 150.
[15] Aggarwal, Charu C., and ChengXiang Zhai. "A survey of text classification algorithms." Mining text data. Springer, Boston, MA, 2012. 163-222.
[16] Zhang, Jiong, et al. "Fast multi-resolution transformer fine-tuning for extreme multi-label text classification." Advances in Neural Information Processing Systems 34 (2021).
[17] Tezgider, Murat, Beytullah Yildiz, and Galip Aydin. "Text classification using improved

ISSN 2616-5767 Vol.5, Issue 1: 53-62, DOI: 10.25236/AJETS.2022.050111

bidirectional transformer." Concurrency and Computation: Practice and Experience (2021): e6486.
[18] Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv: 1810.04805 (2018).