A Deep Learning Method Based Self-Attention and Bi-directional LSTM in Emotion Classification

Rong Fei[1], Yuanbo Zhu[2], Quanzhu Yao[1], Qingzheng Xu[3], Bo Hu[4]

[1] College of Computer Science and Engineering, Xi'an University of Technology, China
[2] China Railway First Survey and Design Institute, China
[3] College of Information and Communication, National University of Defense, China
[4] Beijing Huadian Youkong Technology Co., Ltd, China
annyfei@xaut.edu.cn, 245638830@qq.com, qzyao@xaut.edu.cn, xuqingzheng@hotmail.com, jobobo@outlook.com

Abstract

Traditional recurrent neural network cannot achieve parallelism, while convolutional neural network cannot be used to process variable-length sequence samples directly. In this study, we combined the bidirectional short-time memory (Bi-LSTM) model with the self-attention to form the SA-BiLSTM method, to further improve the performance of the emotion classification model. The SA-BiLSTM method obtains the attention probability distribution by calculating the correlation between the intermediate state and final state. The SA-BiLSTM method weights the state of each moment differently to ensure that the problem of information redundancy is solved while retaining valid information and the accuracy of text classification is improved by optimizing the text feature vector. Experimental results on three different data sets show that the performance of SA-BiLSTM algorithm outperforms the six emotion classification methods by the accuracy, loss rate, time and other performance indicators of the classification model.

Keywords: Sentiment classification, Self-Attention, Deep learning, RNN, Bi-LSTM

1 Introduction

Analysis of text emotional tendency, as an important research focus in the analysis of Internet public opinion, is mainly used to analyse and process subjective information such as attitude, emotion, viewpoint, and tendency in text. Sentiment analysis was first proposed by Pang [1] for the positive or negative classification of movie reviews and Turney [2] for the positive or negative classification of cars and movies in 2002. Subsequent studies on sentiment analysis have been widely carried out in hotels, restaurants, product reviews, Weibo tweets and other fields. Additional developments include positive or negative polarized classification methods [3], five classifications including ratings [4], and eight classifications including specific emotions [5].

Traditional sentiment analysis algorithms are mostly based on shallow machine learning, such as the maximum entropy model [6], conditional random field [7], support vector machine [8], and so on. In 2006, Geoffrey Hinton [9] proposed a method for extracting features to the maximum extent and efficient learning, which has become a hotspot in deep learning research. Many researchers have begun to use deep learning for text sentiment analysis for its' excellent performance in many fields. Due to the long-term dependence of the cyclic neural network on the processing of long text tasks, and considering the temporal information between words in the text, Zhao [10] used the long and short-term memory model (LSTM) for text emotion classification. Kim compared multiple deep learning models on multiple datasets and found that the experimental results of convolutional neural networks was better than other methods [11].

The attention mechanism comes from the human visual attention mechanism, which was first proposed in the field of computer vision [12]. For example, when a human views an image, the main part is usually viewed with high resolution, while the secondary part is viewed with low resolution to reduce interference. The visual attention mechanism helps to focus and quickly understand the image content. Similarly, the attention mechanism can help the algorithm model identify key features and improve training results.

With the continuous development of deep learning technology, neural network models based on attention mechanisms have become a research hotspot in artificial intelligence and natural language processing. Bahdanau [13] was the first to publish research results on attention mechanisms. They introduced an attention mechanism into the Encoder-Decoder [14] machine translation model, which greatly improved the translation effect. Researchers subsequently proposed various improved models for different problems, such

*Corresponding Author: Rong Fei; E-mail: annyfei@xaut.edu.cn
DOI: 10.3966/160792642020092105019

as Multiplicative attention [15], self-attention and key-value Attention [16], and the experimental results exhibited good performance in reading comprehension [17], text implication [18], automatic text summarization [19], etc. Mnih [12] combined the attention mechanism in the cyclic neural network model and achieved good results in the image classification task. Based on the attention mechanism, Chorowski [20] combined the location information and content information of speech to construct a longer end-to-end speech recognition structure than the training corpus. Nakisa [21] proposed a framework of automatic optimization of LSTM super parameters by means of a differential evolutionary algorithm, which was the first systematic study of super parameter optimization in the context of emotion classification, and significantly improved the identification rate and accuracy of four-quadrant emotions.

A self-attention mechanism is a special case of an attention mechanism. That can connect the information of different positions in the input sequence and then calculate a certain expression of the whole sequence. In the text emotion classification task, the self-attention mechanism can learn the word dependence relationship within the sentence, capture the internal structure of the sentence, and improve the accuracy of emotion classification. Self-attention can quickly extract important features of sparse data, reduce dependence on external information, and be better at capturing the internal correlation of data or features. The self-attention mechanism also solves the problems that circulatory neural networks cannot realize parallelism and convolutional neural networks cannot be directly used to deal with variable-length sequence samples.

Based on the advantages of the self-attention mechanism, this paper combines the self-attention mechanism with Bi-LSTM and cross-entropy to construct the SA-BiLSTM model. The SA-BiLSTM model calculates the correlation between the intermediate state and the final state to obtain the probability distribution of attention. The state at each moment is weighted differently, and the problem of information redundancy is solved while retaining valid information. The accuracy of text classification is improved by optimizing the text feature vector.

The rest of this paper is organized as follows: Section 2 introduces the LSTM, Bi-LSTM, attention mechanism, and evaluation indicators of the experiments. Section 3 proposes a SA-BiLSTM model by combining the self-attention mechanism and the Bi-LSTM model to solve the emotion classification problem. The SA-BiLSTM model calculates the correlation between the intermediate state and the final state to obtain the probability distribution of attention. The state at each moment is weighted differently, and the problem of information redundancy is solved while retaining valid information. Section 4 introduces the experiment and related parameters, and section 5 introduces the summary and prospects.

The contributions are summarized as follows:

- We construct a new deep learning method SA-BiLSTM for emotion classification to accelerate the process of classification performance in large data set.
- The proposed SA-BiLSTM calculates the correlation between intermediate state and final state to obtain the probability distribution of attention, weight the state at every moment differently, solve the problem of information redundancy while retaining valid information.
- We design emotion classification experiments in 3 data sets. Results of experiments show that the performance of SA-BiLSTM algorithm outperforms the six emotion classification methods by the accuracy, loss rate, time and other performance indicators of the classification model, and the accuracy of text classification is improved by optimizing the text feature vector.

2 Preliminaries

2.1 Word Vector

2.1.1 Word2Vec Model

Word2Vec obtains word vectors through automatic machine training. It can quickly train text words into word vectors of a specified dimension. Word2Vec's core idea is to build a three-layer neural network to train the probability of a certain word in the corpus dictionary. The network model is divided into two types: CBOW (Continuous Bag-Of-Words) and Skip-gram (Continuous Skip-gram) models. Tomas Mikolov [22] published relevant papers in 2013. The schematic diagrams of the models are shown in Figure 1 and Figure 2. These two models are similar to the traditional neural network structure, i.e., a three-layer network structure.

Word2Vec proposes two optimization schemes for CBOW and Skip-gram models respectively, based on hierarchical softmax and negative sampling methods. For the first optimization scheme, this paper uses the CBOW model as an example, and defines the entire corpus sample set as $Context(w), w$ where w is composed of n words. In the input layer, $V(Context(w)_i)$ is selected to predict the probability of the word $w(t)$. The influence of word order is not considered in the CBOW model, so the word vectors in the input layer are directly summed, as shown in Equation 1:

$$S(w) = \sum_{i=1}^{2n} V(Context(w)_i) \quad \textbf{(1)}$$

Then, the vectors are input into the output layer, and the output layer inputs these vectors into the binary tree

to construct a Huffman tree. The words appearing in the corpus are the leaf nodes of the binary tree, and the word frequency is the weight of the tree. In Huffman coding, 1 is a positive class, and 0 is a negative class. Therefore, for any word w in the corpus sample, there must be a path pw from the root node to the word w in the Huffman tree. There are $lw-1$ branches on the path, which are transformed into binary classification problems and obtained by Logist regression. The expression is shown in Equation 2.

$$P(w \mid Context(w)) = \prod_{j=2}^{l^w} p(d_j^w \mid x_w, \theta_{j-1}^w) \tag{2}$$

$$p(d_j^w \mid x_w, \theta_{j-1}^w) = \begin{cases} \sigma(x_w^T \theta_{j-1}^w), d_j^w = 0 \\ 1-\sigma(x_w^T \theta_{j-1}^w), d_j^w = 1 \end{cases} \tag{3}$$

$$\sigma(x_w^T \theta_{j-1}^w) = \frac{1}{1+e^{-x_w^T \theta}} \tag{4}$$

$$J(\theta) = \frac{1}{n} \sum_{i=1}^{n} \sum_{\substack{-c \le j \le c, \\ j \ne 0}} \log P(w_t \mid w_{t+1}) \tag{5}$$

x_w represents the sum of the sample word vectors. Equation 4 is substituted into the log-likelihood Equation 5, and the optimal solution is obtained by using the stochastic gradient ascent method. The gradient update formula is as shown in Equation 6:

$$\theta_{j-1}^{w^{new}} = \theta_{j-1}^{w^{old}} + \eta[1-d_j^w - \sigma(x_w^T \theta_{j-1}^{w^{old}})]x_w \tag{6}$$

The negative sampling optimization scheme can improve the training speed of the model. Compared with hierarchical softmax, negative sampling adopts a simple random negative sampling method, which reduces the complexity of the model and improves the training speed. Taking the CBOW model as an example, the context of word w is defined as $Context(w)$, so the positive sample of word w is 1, and the negative sample is 0. A number of samples are randomly selected from the negative sample set to form the sample set; thus, the prediction word w is transformed into a classification problem for the two sample sets. This method only needs to traverse the randomly sampled sample set instead of the entire sample space. Although this method can greatly reduce the amount of calculation in the training process, it relies too much on the extraction of negative samples, and the accuracy still needs to be improved when the sample space is large.

2.2 Foundation of a Deep Neural Network Model

LSTM and Bi-LSTM Model

The cyclic neural network processing method lacks the long-term dependence ability. When the sequence information is processed to the later period, there is little information left for the previous period. The standard RNN only remembers the short-distance historical information. Hochreiter proposed the Long Term Short Term (LSTM) model [23] to solve this problem.

The model modifies the internal structure of the memory unit in the standard RNN and adds three thresholds to selectively forget or remember information. The internal structure of the memory unit is shown in Figure 1. The box to the left of Sigmoid represents the forgetting gate, which controls the information retained from the previous state c_{t-1}; the middle box represents the input gate, which controls the information retained from the current input; the box on the right indicates the output gate, which controls the output information.

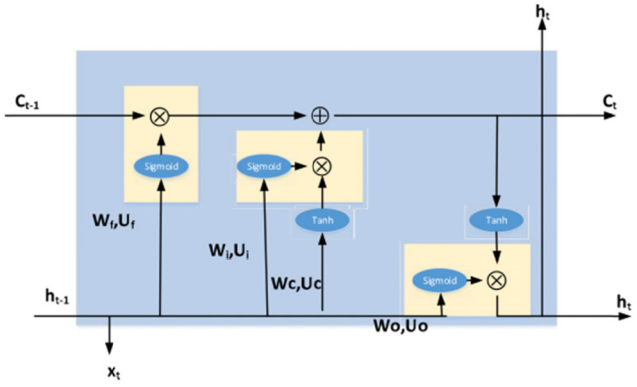


Figure 1. Structure of the LSTM memory unit

Step1: Calculate the forgetting threshold f_t:

$$f_t = Sigmoid(W_t x_t + U_f h_{t-1} + b_f) \tag{7}$$

Step2: Calculate input threshold i_t:

$$i_t = Sigmoid(W_t x_t + U_i h_t + b_i) \tag{8}$$

Step3: Calculate hidden layer state c_t:

$$c_t = f_t \times c_{t-1} + i_t \times Tanh(W_c x_t + U_c h_{t-1} + b_c) \tag{9}$$

Step4: Calculate the output threshold o_t and the output value h_t of the hidden layer in this cycle:

$$o_t = Sigmoid(W_o x_t + U_o h_{t-1} + b_o) \tag{10}$$

$$h_t = o_t \times Tanh(c_t) \tag{11}$$

Similar to convolutional neural networks, fully connected hidden layers are used to adjust the data form; the dimensions of the output layer are mapped to labels for classification.

A large amount of text feature information is hidden in the context. An LSTM network can only deal with one-way time series, and cannot meet the requirements

of text sentiment classification, so it is necessary to extract the following information features. Schuster [24] proposed the concept of Bi-directional Long Short-Term Memory (Bi-LSTM) as shown in Figure 2:

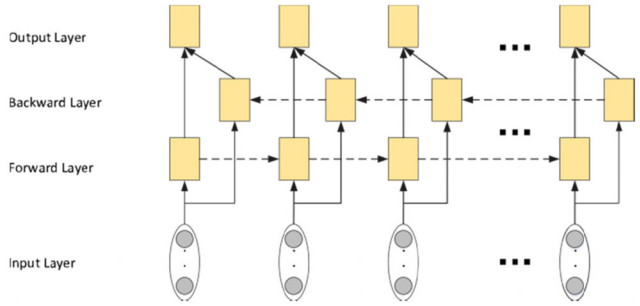


Figure 2. Theoretical structure of Bi-LSTM

Different from one-way propagation based on LSTM, this model contains two independent hidden layers with opposite propagation directions. For the same input data, two eigenvectors regarding the input information, are obtained. Then, a vector is output by stitching or averaging. Therefore, the Bi-LSTM model can better capture bi-directional semantic dependencies.

2.3 Attention Mechanism

The attention mechanism was first used in Encoder-Decoder [20] for natural language processing tasks, such as machine translation or human-machine dialogue. The attention mechanism simulates the human attention mechanism. For the information to be processed, people will focus on a few key information points instead of distributing their attention equally among all the information. The attention mechanism is introduced into the feature extraction and classification model to give different weights to the data in the model, and the information value with higher weight is emphasized in the classification, to improve the classification result.

In the encoder-decoder process, the Encoder model extracts intermediate semantic coding from the input sequence, which contains all information based on input data. Then the Decoder model decodes based on intermediate semantic coding, and outputs the historical information of the sequence, finally splicing to obtain a sequence output. The Encoder-Decoder network model for processing serialized information based on RNN is shown in Figure 3.

The model inputs serialized data and, after the encoding and decoding process, outputs serialized data. During the encoding process, the input data is obtained through the circular calculation of the RNN unit, and the new output information is predicted through the historical output information during the decoding process, which increases the influence of the intermediate semantics on the decoding.

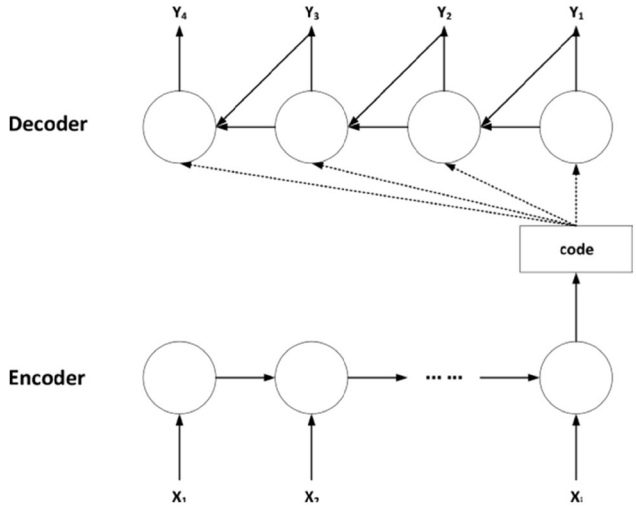


Figure 3. Schematic diagram of the RNN-based Encoder-Decoder model

In the Encoder-Decoder model, the input data is defined as $X=(x_1, x_2, x_3, \ldots, x_i)$, and the output data is $Y=(y_1, y_2, y_3, \ldots, y_i)$; C is the intermediate semantics after encoding, and the process of the Encoder is expressed as Equation 12:

$$C = E(x_1, x_2, \ldots, x_i) \tag{12}$$

The decoder process the intermediate semantic C, which is expressed as Equation 13:

$$y = D(C, y_1, y_2, \ldots, y_{j-1}) \tag{13}$$

The output y_i is calculated from the intermediate semantic encoding c and the decoded historical output values $y_1, y_2, y_3, \ldots, y_{i-1}$.

In the Encoder-Decode model, the same intermediate semantically encoded decoding source causes any element in the input sequence of the model to have the same effect on each element in the output sequence. In addition, the influence of input data on intermediate semantic coding will gradually increase as the sequence spreads. Therefore, the Encoder-Decoder model is affected by the sequence, which does not match the actual output sequence.

Therefore, an attention mechanism is proposed. The attention mechanism puts attention in different positions when processing different information and adds the probability distribution of attention based on input sequence elements in the Encoder-Decoder model. The attention probability distribution controls the influence of input sequence elements on output sequence elements, while retaining valuable information, and reducing irrelevant or weakly related effects. Figure 4 shows the Encoder-Decoder model based on the attention mechanism.

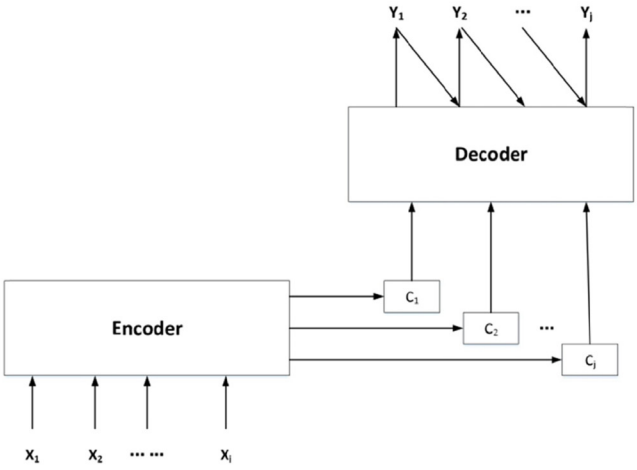


Figure 4. Structure of the Encoder-Decoder model based on the attention mechanism

In the encoder-decoder model based on the attention mechanism, the process of obtaining output data y_j by decoding is shown in Equation 14:

$$y_j = D(C_j, y_1, y_2, \ldots, y_{j-1}) \tag{14}$$

Different from the single intermediate semantic coding in Equation 13, the Encoder-Decoder model based on the attention mechanism produces several different intermediate semantic coding, whose lengths are equal to the number of output elements. C_j is obtained by changing the input sequence through the nonlinear function, and the constituent elements are weighted and summed. The formula is shown in Equation 15:

$$C_j = \sum_{i=1}^{T} a_{ij} S(x_i) \tag{15}$$

The input data changes the function to obtain $S(x_i)$, and the number of elements in the input sequence is T. a_{ij} represents the attention probability distribution value of the input x_i to the output y_j. Attention-based intermediate semantic coding is obtained through the probability distribution of attention.

The key of the Encoder-Decoder model based on the attention mechanism is to calculate the probability distribution of attention. The core idea of the attention probability distribution calculation method is shown in Figure 5. The hidden layer semantics based on the output data and the hidden layer semantics of the input information are compared to form a probability distribution.

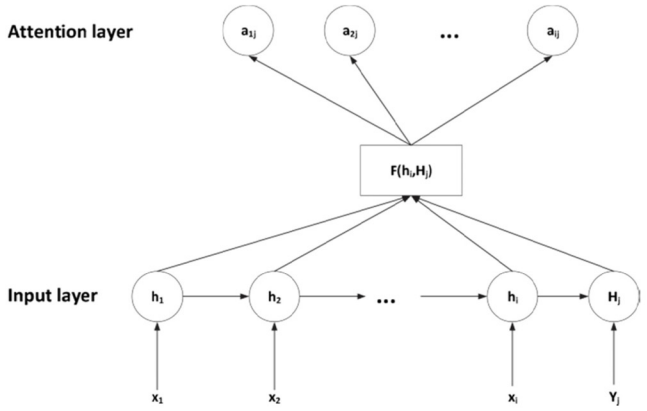


Figure 5. Calculation of the attention probability distribution

The attention probability distribution a_{ij} of output y_j and input x_i is expressed as Equation 16:

$$a_{ij} = F(h_i, H_j) \tag{16}$$

h_i represents the state of input x_i in the hidden layer of the Encoder, H_j represents the state of output y_j in the hidden layer of the Decoder, and F calculates the probability that the two states match.

The attention mechanism's processing of text data can extract better text features, and identify key parts of the text through the value of attention, making it more interpretable.

3 Self-attention-BiLSTM

3.1 Attention Mechanism

The kernel mechanism of the attention mechanism is shown in Figure 6. The source is composed of a series of <Key, Value> data pairs. A key value query has three basic elements: Query, Key and Value. First it calculates the similarity between the Query and Key to obtain the weight coefficient of the Value corresponding to each Key, then weights it to obtain the weight and Value of the key, and finally obtains the Attention value. The attention mechanism is essentially a weighted sum of the Values of elements in the Source, while the Query and Key calculate the weight coefficient of the corresponding Value. The kernel mechanism formula is shown in Equation 17.

$$\begin{aligned} &Attention\,(Query, Source) \\ &= \sum_{i=1}^{L_x} Similarity(Query, Key_i) * Value_i \end{aligned} \tag{17}$$

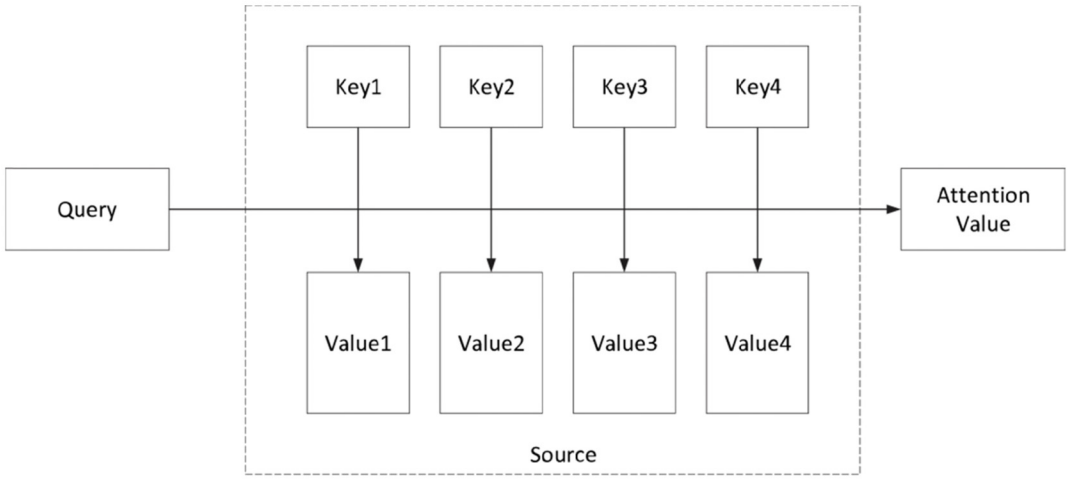


Figure 6. Kernel mechanism of the attention mechanism

$L_x = \| Source \|$ represents the length of the data source.

The calculation process of attention is abstracted into three steps shown in Figure 7:

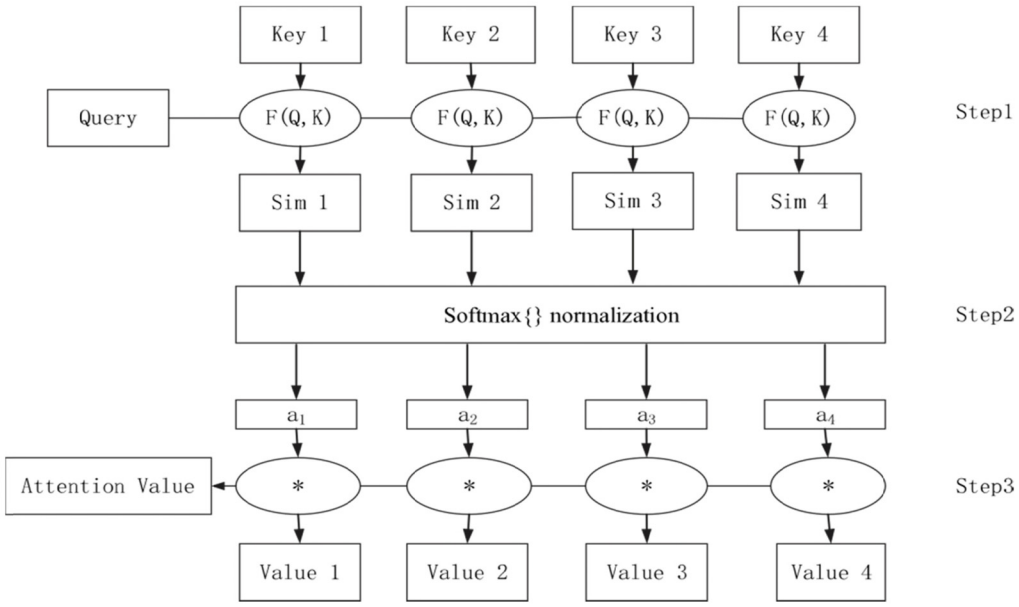


Figure 7. Calculation process of the attention mechanism

(1) Calculate the weight coefficients based on the Query and Key, and calculate the similarity or correlation between the two based on the Query and Key.

(2) Normalize the original scores.

(3) Obtain a weighted summation of Value according to the weight coefficient.

K (Key) represents the keyword, Q (Query) represents the Query, F represents the function, V (Value) represents the weight Value, Sim represents the similarity, a represents the weight coefficient, and A represents the Attention Value. When a statement is entered, each word is calculated with all words separately.

According to the Query and a certain keyword (K_i), different functions and calculation mechanisms are introduced to calculate the similarity or correlation between the two. The three common formulas are as follows:

$$Sim(Q, K_i) = Q \cdot K_i \tag{18}$$

$$Sim(Q, K_i) = \frac{Q \cdot K_i}{\| Q \| \cdot \| K_i \|} \tag{19}$$

$$Sim(Q, K_i) = MLP(Q, K_i) \tag{20}$$

Equation 18 the dot product method, Equation 19 shows the cosine similarity method, and Equation 20 shows the method of evaluation through a neural network (MLP).

In the first step, different values are generated according to different methods. The scores of the

previous step are numerically converted in the second stage. Normalization can convert the original score into a probability distribution of element weights; the internal mechanism of softmax can highlight the weights of important elements. Equation 21 is used to calculate the weight coefficient a_i corresponding to $Value_i$:

$$a_i = softmax(Sim_i) = \frac{e^{Sim_i}}{\sum_{j=1}^{L_x} e^{Sim_j}} \tag{21}$$

Equation 22 obtains the attention value by weighted summation.

$$A(Q, S) = \sum_{i=1}^{L_x} a_i \cdot Value_i \tag{22}$$

The self-attention mechanism is also called intra-attention. In the general Encoder-Decoder framework, the input Source and output Target content are inconsistent. For instance, in the English-Chinese machine translation, the Source is English and the Target is Chinese. The attention mechanism occurs between the Target element Query and all elements of the Source. The self-attention mechanism occurs between internal elements of the Source or internal elements of Target, which can also be understood as the case where $Target = Source$.

The self-attention mechanism can capture syntactic or semantic features between words in a sentence. The cyclic neural network or LSTM model is calculated sequentially. Long-distance interdependent features require multi-step information accumulation. The longer the distance is, the less likely it is to be captured effectively. The self-attention mechanism is better able to capture long-distance interdependence features in the sentence, which can directly connect any two words in the sentence instead of calculating according to the sequence, effectively shortening the distance between the long-distance-dependent features and increasing the parallelism of the calculation.

3.2 Position Embedding

The self-attention mechanism cannot capture sequence information. Order is important in time series and NLP tasks, representing local or global structures. In the self-attention mechanism, the word order of the sentence is shuffled, and the Key and Value are shuffled in line order. The result of attention is unchanged. To make use of the order information of the input sequence, the absolute position or relative position of the input sequence is introduced. This paper uses positional encoding to encode the input, and the vector dimension after encoding is d_{model}. Specifically, the sine and cosine function codes shown in Equation 23 are used to encode.

$$\begin{cases} PE_{2i}(p) = \sin(p/10000^{2i/d_{pos}}) \\ PE_{2i+1}(p) = \cos(p/10000^{2i/d_{pos}}) \end{cases} \tag{23}$$

Equation 23 maps the position of $id = p$ to a position vector of d_{pas} dimension, and the value of the ith element of the vector is $PE_i(p)$.

3.3 Complexity Analysis

Table 1 gives a comparison of the self-attention mechanism, recurrent neural network, and convolutional neural network in three aspects:

(1) Time complexity of each layer;
(2) Whether it can be parallel;
(3) Maximum path length.

Table 1. Comparison of complexity results

Layer type	time complexity	parallel complexity	Maximum path length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(K \cdot n \cdot d^2)$	$O(1)$	$O(\log_k(n))$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

As seen from Table 1, when the input sequence n is smaller than the representation dimension d, the time complexity of each layer is the smallest in the attention mechanism. When n is large, one solution is a restricted self-attention mechanism, that is, no word calculates Attention with all words, and instead, each word calculates Attention with limited r words. In terms of parallelism, the self-attention mechanism and the convolutional neural network can be well parallelized, and do not depend on the previous calculation, which is better than the recurrent neural network. In long-distance dependence, since the self-attention mechanism calculates Attention for every word and all words, no matter how long the distance is between them, the maximum path length is only 1, which can capture long-distance dependencies.

3.4 General Structure of the SA-BiLSTM Model

The attention mechanism was originally used in Encoder-Decode to emphasize the different influence of different input data on output data. Based on the idea of Attention, the migration is applied to the text feature extraction and classification model based on Bi-LSTM, which can also be used to calculate the different influence of the classification result on the input data, thus optimizing the feature vector of the text and improving the classification performance.

This study combines the self-attention mechanism with Bi-LSTM to construct the SA-BiLSTM model. The key of the model is to calculate the probability

distribution of attention. Unlike the traditional Bi-LSTM, which directly combines the states of two layers as the final feature, the SA-BiLSTM model combines the states of each moment as the final state to calculate the probability distribution of the attention to the final state at each moment, thus using attention distribution to further optimize the final state as the final text feature.

The SA-BiLSTM model mainly includes three parts; the first is the construction of word vectors, the second is feature extraction, and the third is a classifier.

Regarding the construction of the word vector, this model uses Word2Vec technology to obtain the word vector containing semantic information through pre-training; as for the classifier, the model directly uses the softmax function to directly connect with the output of the model, and the feature extraction is the focus of this model. In the feature extraction part of the model, based on Bi-LSTM, it is known that LSTM can effectively process the sequence information of similar texts. Bi-LSTM also references the overall context information of the text at the same time, which can be more comprehensive. On this basis, the self-attention mechanism is added to enable the model to focus on retaining key information while ensuring comprehensiveness.

The expression of the objective function of the SA-BiLSTM model is shown in Equation 24, using softmax as the output layer normalization calculation, and combining the cross-entropy loss function.

$$L = \sum_{i=1}^{T} Y_i \log(y_i) \tag{24}$$

where T represents the amount of text in the corpus, Y_i represents the true probability distribution vector of the current text category, y_i represents the probability distribution vector of the current text predicted by the classification model, and the dimensions of the vector are equal to the number of classification labels. By minimizing the objective function, the classification model can be trained.

In Figure 8, M represents the sum of the final hidden-layer state values in the independent directions in the Bi-LSTM, which is called the final state of the Bi-LSTM; D represents the attention probability distribution of the state of the hidden layer unit to the final state at any time, where the component d_n represents the attention probability of the Bi-LSTM state an at time n to the final state and a_n is obtained by adding the states in the independent directions at that moment; S_a represents the text feature vector after attention weighting.

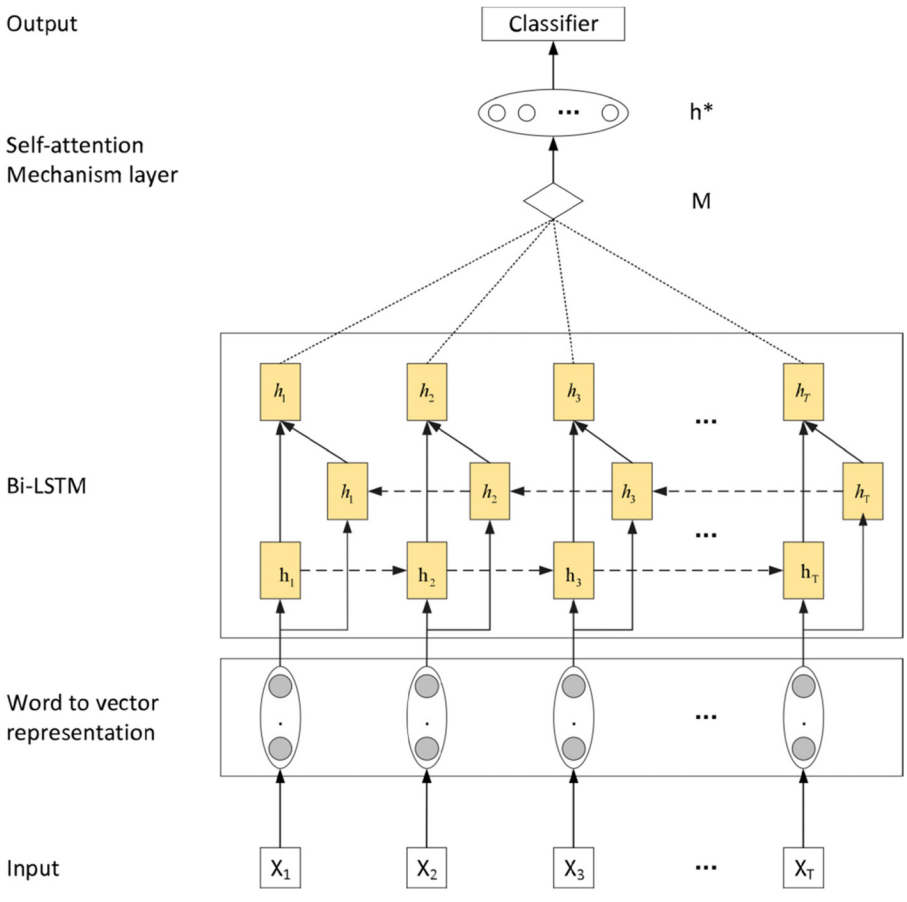


Figure 8. SA-BiLSTM model

Models based on the attention mechanism generally include two parts of the calculation process, one is the calculation process of the attention probability distribution, and the other is the final feature calculation process based on the attention distribution. In this model, the attention probability of the output data at time n for the final state is calculated as follows:

$$D_n = \frac{\exp(a'_n)}{\sum_{i=1}^{N} \exp(a'_i)} \tag{25}$$

$$a'_n = a_X^T UF \tag{26}$$

The formula uses the softmax function as the calculation method of the attention probability distribution, where N represents the number of input sequence elements; U is the weight matrix; F is the sum of the final hidden layer state values in each independent direction in Bi-LSTM; and a_n represents the sum of the bidirectional hidden-layer state values at time n. The calculation method is shown in Equation 27.

$$a_n = Tanh(Uw_n + Va_{n-1}) \tag{27}$$

where w_n represents the input data of the model at time t_n; a_{n-1} represents the state of the hidden layer of the model at the previous moment of t_n; and U and V respectively represent the weight parameter matrix of the model input layer and the hidden layer. In this model, the calculation process based on the final feature S_a of the self-attention distribution is expressed as:

$$Sa = \sum_{n=1}^{N} D_n a_n \tag{28}$$

where N represents the number of input sequence elements, D_n represents the attention probability of the output data at time n for the final state; and a_n represents the sum of the hidden-layer states of two independent directions at time n. After obtaining the text feature vector S_a based on the self-attention mechanism, the probability distribution of the classification labels is calculated through the softmax function of the output layer, and the calculation process is expressed as:

$$y = softmax(Sa') = \frac{\exp(Sa'_i)}{\sum_{j=1}^{T} \exp(Sa'_j)} \tag{29}$$

$$Sa' = VSa \tag{30}$$

where T is the number of category labels; V represents the weight matrix of the model output layer; Sa'_i represents the i-th component value in the vector Sa'; and the length of the vector is equal to the number of classification labels. Through the classification by the softmax function, the probability distribution y of the text category based on the self-attention mechanism can be obtained, through which, combined with the real category distribution Y, the cross-entropy loss can be obtained and expressed as:

$$E(Y, y) = -Y \log(y) \tag{31}$$

where Y represents the probability distribution of the true category; y represents the probability distribution of the category predicted by the model.

4 Experiment and Analysis

4.1 Experimental Setting and Result Analysis

4.1.1 Experimental Environment

The experimental environment configuration is shown in Table 2.

Table 2. Experimental environment and configuration

Experimental environment	Specific configuration
Operating system	Windows 10 (64-bit)
CPU	Intel(R) Core(TM) i7-4790, 3.6GHz
Memory	8G
Hard disk	1T
Programming language	Python 3.5
Deep learning framework	keras 2.0
Word vector training tool	Word2Vec

4.1.2 Dataset

Since the classification model could have different adaptability to texts of different lengths, to verify the performance of the model, the experiments were tested with different types of datasets from multiple well-known corpora. This experiment used three types of text data, covering different lengths and different types of text classification tasks.

Table 3 descripts these three datasets.

Table 3. Dataset introduction

Dataset	Number of categories	Training Testing set	Average length
SST2	2	6920/2693	18
SUBJ	2	9000/1000	21
IMDB	2	25000/25000	294

SST2: Sentiment Treebank Stanford sentiment classification dataset proposed by Socher et al. [25]. Each sample is a movie review with two predefined categories: positive and negative.

SUBJ: A subjective / objective binary classification dataset proposed by Pang et al. [26], whose predefined

categories include subjective and objective.

IMDB: Emotional binary classification dataset proposed by Maas et al. [3], whose predefined categories are positive and negative. Compared with the first three datasets, each sample of IMDB contains many sentences, whose average length is much larger than that of the first three datasets.

4.1.3 Evaluation Index

Assuming that the classification targets are positive and negative, the chose evaluation terms are as follows:

(1) Accuracy

$Accuracy = (TP + TN)/(P/N)$, which is the number of correctly classified samples divided by all samples. The higher the accuracy, the better the classifier;

(2) Loss Function

The loss function is used to evaluate the difference between the predicted value and the true value of the model. In classification or regression, the loss function is typically selected as the objective function of the algorithm. Generally the smaller the loss function is, the better the model can reflect real data. This paper uses the cross-entropy loss function in deep learning models.

4.2 Reference System and Hyper-parameters Settings

To test the effect of the mode on sentiment classification, this study conducted comparative experiments on the following benchmark models:

- **LSTM:** proposed by Hochreiter et al. [25], this model mainly solves the problems of the lack of long-term dependence and the disappearance of gradients in the recurrent neural network processing method.
- **Bi_LSTM:** Bi-directional Long Short-Term Memory neural network was proposed by Schuster et al. [24]. This model differs from LSTM with one-way propagation in the hidden-layer in that it contains two independent hidden-layers with opposite propagation directions. Therefore, for the same input data, two hidden-layer outputs can ultimately be obtained.
- **CNN:** proposed by Hubel et al., the model is mainly divided into four parts: the lookup layer, convolution layer, pooling layer and softmax layer.
- **CNN-RNN:** proposed by Hsu ST, Moon C [27] et al., the model is mainly divided into three parts. The first part obtains the phrase representation through CNN. The second part obtains the sentence representation through two-way RNN. The last part obtains the final representation of the sentence by averaging, and then softmax is used to obtain the category of the sentence.
- **Self-Attention:** The self-attention mechanism was proposed by the Google Machine Translation team in 2017 [28-30]. The model can link information at different positions of the input sequence, and calculate a certain expression of the entire sequence.
- **SA-PE:** Self Attention with **Position** Embedding added [31-32].

The hyper-parameters configuration of the models proposed in this paper are shown in Table 4, and the hyper-parameters settings of each model are shown in Table 5.

Table 4. Model hyper parameter configuration

Name	Meaning
Batch_size	number of batch training samples
Hidden_dam	hidden layer node
Embedding_dam	word vector dimension
Filters	filter size
Kernel_size	number of convolution kernels
Max_features	the maximum number of features of the bag of words model
Epoch	times of model iterations
Min_count	lowest word frequency

Table 5. Model hyper parameter configuration

	LSTM	Bi_LSTM	CNN	CNN-RNN	Self Attention	SA-PE
Hidden_dam	256	256	256	256	256	256
Batch_size	32	32	32	32	32	32
Embedding_dam	50	50	50	50	50	50
Filters	-	-	250	250	-	-
Kernel_size	-	-	3	3	-	-
Max_features	5000	5000	5000	5000	5000	5000
Epoch	10	10	10	10	10	10
Min_count	10	10	10	10	10	10
Dropout	0.2	0.2	0.2	0.2	0.5	0.5
Activation function	Sigmoid	Sigmoid	Relu	Relu/Sigmoid	Sigmoid	Sigmoid

This part verifies the effectiveness of the SA-BiLSTM model in sentiment classification tasks and then compares and analyses the benchmark model in accuracy, loss rate, and time performance in three datasets.

Analysis of experimental results of SA-BiLSTM model comparison are as follows:

To verify the effectiveness of the SA-BiLSTM model, it was compared to and analysed with the results of the emotion classification task in different datasets with the benchmark model, in which the benchmark model and parameter settings are described in section 4.2.

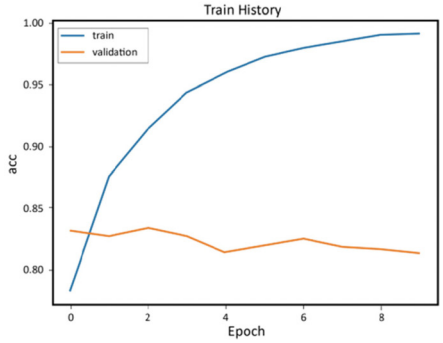
It can be seen from Table 6 that the experimental results of different datasets are different. In this paper, the IMDB dataset with the largest amount of data is selected for the following specific analysis:

Table 6. Results of different Dropout values

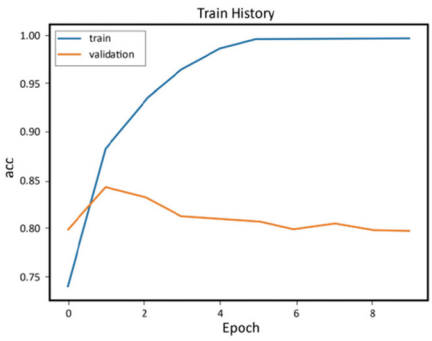
	IMDB			SST2			SUBJ		
	Accu -racy (%)	Loss rate	Time (s)	Accu -racy (%)	Loss rate	Time (s)	Accu -racy (%)	Loss rate	Time (s)
LSTM	80.98	0.8170	1187	78.43	0.7562	1048	-	-	-
Bi-LSTM	82.99	0.2665	1032	80.02	0.2445	965	81.49	0.2431	985
CNN	88.34	0.5588	1374	86.24	0.5632	1211	88.24	0.4752	1279
CNN-RNN	83.31	0.8020	1426	-	-	-	-	-	-
Self Attention	77.46	0.3860	649	76.32	0.3546	621	76.92	0.3655	630
SA-PE	80.95	0.3138	675	78.48	0.3271	654	79.84	0.3248	654
SA-BiLSTM	88.91	0.2874	1130	86.75	0.2791	1024	88.38	0.2776	1049

4.2.1 Accuracy Analysis

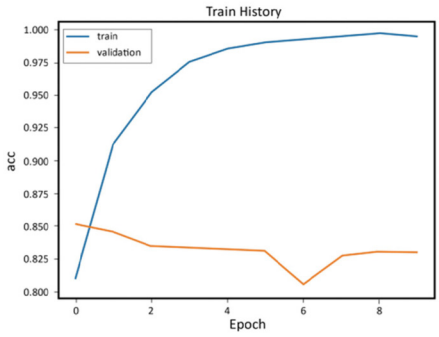
Figure 9 show the performance of each model on the training set and test set in the emotion classification task. The horizontal axis represents the number of iterations, and the vertical axis represents the accuracy. The blue curve represents the change in accuracy in the training set, and the orange curve represents the change in accuracy in the test set. Figure 10 shows the change in accuracy of each model within 10 iterations. Figure 11 shows the change in accuracy of the SA-PE and Self-Attention models within 10 iterations. From the experimental results above, the following can be found:



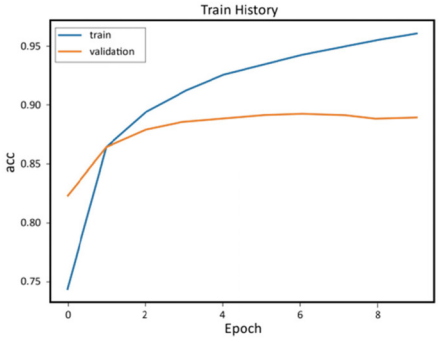
(a) LSTM



(b) SA-PE



(c) Bi-LSTM



(d) SA-BiLSTM

Figure 9. Changes in the accuracy of the LSTM, Bi-LSTM, Self Attention-P and SA-BiLSTM models

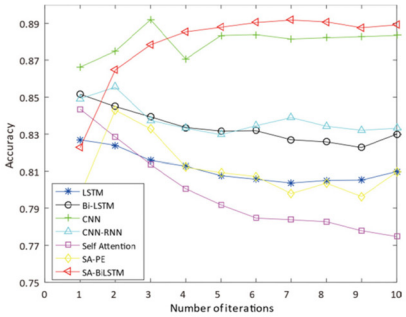


Figure 10. Variations in the accuracy of each model of different iterations

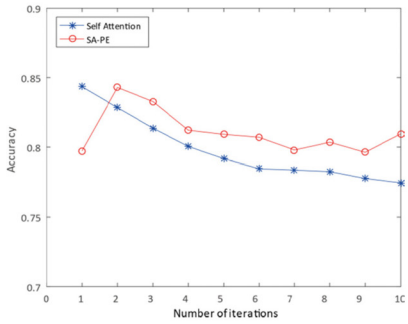


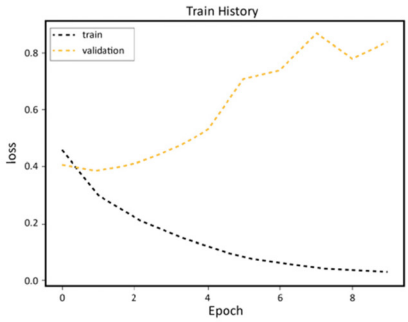
Figure 11. Changes in accuracy of Self Attention and SA-PE models of different iterations

(1) As seen from Figure 8, after 10 iterations, the SA-BiLSTM model proposed in this paper has the highest accuracy of 88.91%, and the Self Attention model has the lowest accuracy of 77.46%. The accuracies of the SA-PE and LSTM models are close; the accuracy rates of the CNN-RNN and Bi-LSTM models are close; and the accuracy rates of the SA-BiLSTM and CNN models are close. After the fourth iteration, the accuracy of SA-BiLSTM is higher than that of the other models', and the accuracy fluctuates slightly.

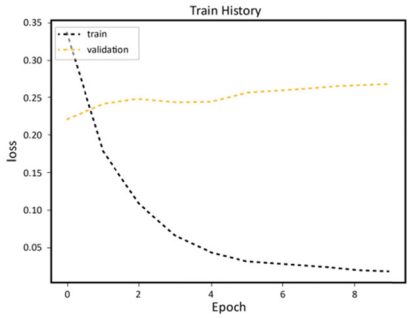
(2) As seen from Figure 9, the SA-BiLSTM model has higher accuracy than the Bi-LSTM model. Both models use the Bi-LSTM neural network layer to extract the key information of the text. The difference between the two models is that SA-BiLSTM adds a Self Attention layer to the Bi-LSTM layer and uses the Self Attention layer to assign corresponding weights to highlight the key features of the text. The final accuracy is 5.92% higher than that of the Bi-LSTM model. Compared with the SA-PE model, the SA-BiLSTM model adds a Bi-LSTM layer to extract text features, and the final accuracy is improved by 7.96%. The reason is that the advantage of the Self Attention layer is that it can capture the global connection within one step, because it directly compares the sequences pair by pair, but it cannot model the position information well. Therefore, the SA-BiLSTM model combined with the BiLSTM layer is more sufficient to obtain global information and features, can fully integrate their respective advantages, and can improves the accuracy of emotion classification.

4.2.2 Loss Rate Analysis

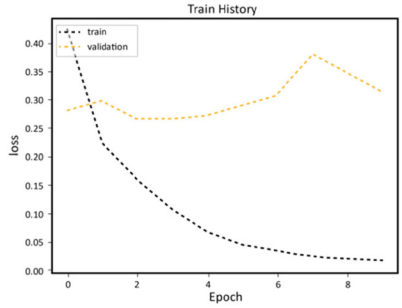
Figure 12 shows the performances of the loss rates of the LSTM, Bi-LSTM, SA-PE, and SA-BiLSTM models in sentiment classification tasks on training and testing sets. The horizontal axis represents the number of iterations, and the vertical axis represents the loss rate. The black curve represents the change in loss rate on the training set, and the yellow curve represents the change in loss rate on the test set. Figure 13 shows the change in the loss rate of each model within 10 iterations. From the experimental results above, the following can be found:



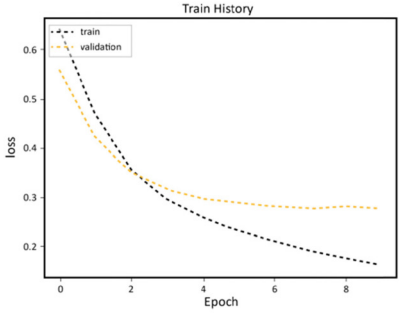
(a) LSTM



(b) Bi-LSTM



(c) SA-PE



(d) SA-BiLSTM

Figure 12. Loss rate changes of the LSTM, Bi-LSTM, SA-PE, SA-BiLSTM models

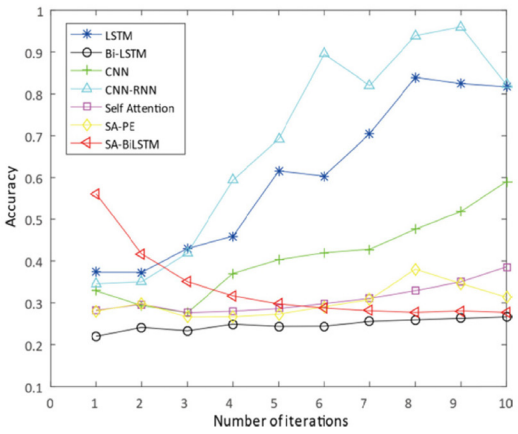


Figure 13. Accuracy rate change of each model with different number of iterations

(1) It can be seen from Figure 12 that the loss rate of the four models of LSTM, Bi-LSTM, SA-PE, and SA-BiLSTM on the training set decreases as the number of iterations increases and eventually stabilizes. However, the loss rate on the test set varies greatly. The final loss rate of the Bi-LSTM model after 10 iterations is 0.2665, and the loss rate of the SA-PE model after 10 iterations is 0.3138. SA-BiLSTM adds a Self Attention layer to the Bi-LSTM layer, and its model has a loss rate of 0.2874 after 10 iterations, which is 0.0209 more than that of the Bi-LSTM model and 0.0264 less than that of the SA-PE model. The loss rate of the three models remained at a low level and stabilized after 10 iterations.

(2) As seen from Figure 13, the CNN-RNN model has the highest loss rate after 10 iterations, and the Bi-LSTM model has the lowest loss rate. With the increase in the number of iterations, the loss rate of the SA-BiLSTM model proposed in this paper continues to decrease and eventually converges. The loss rate changes of the Self-Attention model and the SA-PE model are relatively close, and the final difference is 0.0722. The analysis in this paper is that the SA-BiLSTM model with the self-attention mechanism added solves the problem of long-range and global dependencies in sentiment classification tasks, making the model's generalization ability better.

4.1.3 Time Performance Analysis

Figure 14 shows the time consumption of each model in the sentiment classification task. As seen from the figure, the Self Attention model takes the shortest time, which is 649 seconds; the SA-PE model with position embedding takes 675 seconds; the SA-BiLSTM model proposed in this chapter takes 1130 seconds. From the literature [28], it can be found that the time complexity of the Self Attention layer is $O(n^2 \cdot d)$; the time complexity of the recurrent neural network layer is $O(n \cdot d^2)$; and the time complexity of the convolutional neural network layer is $O(k \cdot n \cdot d^2)$. When the input sequence n is smaller than the representation dimension d, the time complexity self-attention mechanism of each layer is more advantageous. Therefore, the Self Attention model is better overall in the emotion classification task.

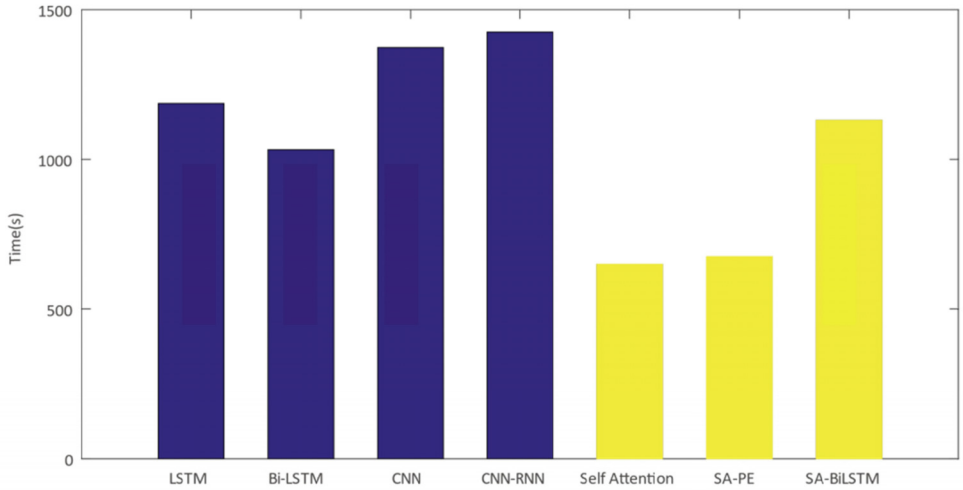


Figure 14. Time analysis of each model

5 Conclusion

This paper mainly introduces the SA-BiLSTM text feature extraction and classification model combined with the self-attention mechanism and analyses the performance of the model through experiments. Firstly, the essential thought and structural principle of the self-attention mechanism are introduced, and the calculation method of the position embedding of the self-attention mechanism is explained. The self-attention mechanism is compared with the recurrent neural network and the convolutional neural network in terms of the time complexity of each layer, whether they can be parallelized, and the maximum path length.

Then, the general structure of constructing a new SA-BiLSTM model by combining the self-attention mechanism with the Bi-LSTM model is introduced. The experiment is divided into two parts. In the first part, the optimal parameter values are selected by comparing the word vector dimensions, the number of iterations, the activation function, and the influence of the four different parameter values on the experimental results. In the second part, the SA-BiLSTM model is compared with the benchmark model, and the performance of the model is analyzed based on the results. It is indicated that the SA-BiLSTM model, on the one hand, eliminates the unreasonable influence of input data on the output data, and emphasizes the role of key information in the classification results and, on the other hand, uses the characteristics of Bi-LSTM to alleviates the problem of different influence on the results due to the different order of input information and effectively improves the accuracy of emotion classification. This paper also introduces the experimental environment, benchmark system, parameter settings, and three types of datasets. It compares six sentiment classification methods via statistical performance indicators consumption such as accuracy, loss rate, and time consumption of the classification model. It also explores the impact of different datasets on sentiment classification.

In the next work, we will consider the characterisics of CNN with its extraction of text features and RNN with its capacity on series tasks, which can be combined with Self-Attention to construct a better model for text feature extraction and classification.

Acknowledgments

This work was supported in part by National Key Research and Development Program of China (2018YFB1201500), Natural Science Foundation of China (61773313), Project of Technological Planning Project of Beilin District of Xi'an(GX1819) and Teaching Research Foundation of Xi'an University of Technology (XJY1866).

References

[1] B. Pang, L. Lee, S. Vaithyanathan, Thumbs Up?: Sentiment Classification Using Machine Learning Techniques, *ACL-02 conference on Empirical methods in Natural Language Processing- Volume 10*, Philadelphia, PA, USA, 2002, pp. 79-86.

[2] P. D. Turney, Thumbs Up or Thumbs Down?: Semantic Orientation Applied to Unsupervised Classification of Reviews, *40th Annual Meeting on Association for Computational Linguistics*, Philadelphia, Pennsylvania, USA, 2002, pp. 417-424.

[3] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, C. Potts, Learning Word Vectors for Sentiment Analysis, *49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies- Volume 1*, Portland, Oregon, USA, 2011, pp. 142-150.

[4] C. Du, L. Huang, Sentiment Analysis Method Based on Piecewise Convolutional Neural Network and Generative Adversarial Network, *International Journal of Computers, Communications & Control*, Vol. 14, No. 1, pp. 7-20, February, 2019.

[5] Y. Zhang, Y. Jiang, Y. Tong, Study of Sentiment Classification for Chinese Microblog Based on Recurrent Neural Network, *Chinese Journal of Electronics*, Vol. 25, No. 4, pp. 601-607, July, 2016.

[6] A. McCallum, D. Freitag, F. C. Pereira, Maximum Entropy Markov Models for Information Extraction and Segmentation, *Seventeenth International Conference on Machine Learning (Icml)*, Stanford, CA, USA, 2000, pp. 591-598.

[7] Y. F. Pan, X. Hou, C. L. Liu, Text Localization in Natural Scene Images Based on Conditional Random Field, *2009 10th International Conference on Document Analysis and Recognition*, Barcelona, Spain, 2009, pp. 6-10.

[8] T. Joachims, *Learning to Classify Text Using Support Vector Machines*, Springer Science & Business Media, 2002.

[9] G. E. Hinton, S. Osindero, Y.-W. Teh, A Fast Learning Algorithm for Deep Belief Nets, *Neural Computation*, Vol. 18, No. 7, pp. 1527-1554, July, 2006.

[10] Y. Zhao, B. Qin, T. Liu, D. Tang, Social Sentiment Sensor: A Visualization System for Topic Detection and Topic Sentiment Analysis on Microblog, *Multimedia Tools and Applications*, Vol. 75, No. 15, pp. 8843-8860, August, 2016.

[11] Y. Kim, Convolutional Neural Networks for Sentence Classification, *2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, 2014, pp. 1746-1751.

[12] V. Mnih, N. Heess, A. Graves, K. Kavukcuoglu, Recurrent Models of Visual Attention, *Advances in Neural Information Processing Systems*, Montréal Canada, 2014, pp. 2204-2212.

[13] D. Bahdanau, K. Cho, Y. *Bengio, Neural Machine Translation by Jointly Learning to Align and Translate*, https://arxiv.org/abs/1409.0473, 2014.

[14] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, *Learning Phrase Representations Using RNN Encoder-decoder for Statistical Machine Translation*, https://arxiv.org/abs/1406.1078, 2014.

[15] D. H. Hubel, T. N. Wiesel, Receptive Fields, Binocular Interaction and Functional Architecture in the Cat'S Visual Cortex, *The Journal of Physiology*, Vol. 160, No. 1, pp. 106-154, January, 1962.

[16] X. Glorot, Y. Bengio, Understanding the Difficulty of Training Deep Feedforward Neural Networks, *Thirteenth International Conference on Artificial Intelligence and Statistics*, Chia Laguna, Sardinia, Italy, 2010, pp. 249-256.

[17] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, Greedy Layer-wise Training of Deep Networks, *Advances in Neural Information Processing Systems*, Vancouver, BC, Canada, 2006, pp. 153-160.

[18] M. A. Ranzato, C. Poultney, S. Chopra, Y. L. Cun, Efficient

Learning of Sparse Representations with an Energy-based Model, *Advances in Neural Information Processing Systems*, Vancouver, BC, Canada, 2006, pp. 1137-1144.

[19] D. Erhan, Y. Bengio, A. Courville, P. A. Manzagol, P. Vincent, S. Bengio, Why Does Unsupervised Pre-training Help Deep Learning?, *Journal of Machine Learning Research*, Vol. 11, pp. 625-660, February, 2010.

[20] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, Y. Bengio, Attention-based Models for Speech Recognition, *Advances in Neural Information Processing Systems*, Montréal, Canada, 2015, pp. 577-585.

[21] B. Nakisa, M. N. Rastgoo, A. Rakotonirainy, F. Maire, V. Chandran, Long Short Term Memory Hyperparameter Optimization for a Neural Network Based Emotion Recognition Framework, *IEEE Access*, Vol. 6, pp. 49325-49338, September, 2018.

[22] T. Mikolov, K. Chen, G. Corrado, J. Dean, *Efficient Estimation of Word Representations in Vector Space*, https://arxiv.org/abs/1301.3781, 2013.

[23] S. Hochreiter, J. Schmidhuber, Long Short-term Memory, *Neural Computation*, Vol. 9, No. 8, pp. 1735-1780, November, 1997.

[24] M. Schuster, *Bi-directional Recurrent Neural Networks for Speech Recognition*, IEICE Technical Report, SP96, October, 1996.

[25] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, C. Potts, Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank, *2013 Conference on Empirical Methods in Natural Language Processing*, Seattle, Washington, USA, 2013, pp. 1631-1642.

[26] B. Pang, L. Lee, A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts, *42nd Annual Meeting on Association for Computational Linguistics*, Barcelona, Spain, 2004, pp. 271-278.

[27] S. T. Hsu, C. Moon, P. Jones, N. Samatova, A Hybrid CNN-RNN Alignment Model for Phrase-aware Sentence Classification, *15th Conference of the European Chapter of the Association for Computational Linguistics*, Vol. 2, Valencia, Spain, 2017, pp. 443-449.

[28] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is All You Need, *Advances in Neural Information Processing Systems*, Long Beach, CA, USA, 2017, pp. 5998-6008.

[29] J. Bobadilla, F. Ortega, A. Gutiérrez, S. Alonso, Classification-based Deep Neural Network Architecture for Collaborative Filtering Recommender Systems, *International Journal of Interactive Multimedia and Artificial Intelligence*, Vol. 6, No. 1, pp. 68-77, February, 2020.

[30] M. Anbarasan, B. A. Muthu, C. B. Sivaparthipan, R. Sundarasekar, S. Kadry, S. Krishnamoorthy, D. J. Samuel, A. A. Dasel, Detection of Flood Disaster System Based on IoT, Big Data and Convolutional Deep Neural Network, *Computer Communications*, Vol. 150, pp. 150-157, January, 2020.

[31] L. Liu, P. P. Wu, X. Z. Wang, Short-time Traffic Flow Prediction Based on Wavelet Neural Network, *Journal of Internet Technology*, Vol. 20, No. 4, pp. 1237-1246, July, 2019.

[32] F. Zhou, J. Liu, T. Chang, Fast Background Subtraction for Microorganism Detection in Wireless Visual Sensor Networks, *Journal of Internet Technology*, Vol. 17, No. 3, pp. 469-481, May, 2016.

Biographies

Rong Fei received the Ph.D. degree in Power Electronic and Electrical Drive from the Xi'an University of Technology, Xi'an, China, in 2009. She is currently an associate Professor. Her main research interests are Community detection, stochastic opposition algorithm, location-based service.

Yuanbo Zhu received his M.S. degree in computer software and theory from XI'AN University of Technology, China, in 2019. Currently he is working at China Railway First Survey and Design Institute. His research interests include DATABASE Technology, Data Mining.

Quanzhu Yao is currently a professor in Department of Computer science and engineering, XI'AN University of Technology. His research interests include DATABASE Technology, Data Mining, etc.

Qingzheng Xu received the B.S. degree in information engineering from the PLA University of Science and Technology, Nanjing, China, in2002, and the Ph.D.degree in control theory and engineering from the Xi'an University of Technology, Xi'an, China, in 2011. From 2002 to 2017, he was a Lecturer with the Xi'an Communications Institute, Xi'an. Since 2018, he has been an Associate Professor with the College of Information and Communication, National University of Defense Technology, Xi'an. He is currently a Visiting Scholar with the School of Computer Science and Engineering, Nanyang Technological University, Singapore. His research interests include opposition-based learning, nature-inspired computation, and combinatorial optimization.

Bo Hu received the M.S. degree in signal and information processing from Xi'an University of Technology, Xi'an, China in 2009. He is currently a Director of R&D with Beijing Huadian Youkong Technology Co., Ltd. His main research interests are Image Processing, location-based service.