Hindawi
Mathematical Problems in Engineering
Volume 2022, Article ID 5909033, 14 pages
https://doi.org/10.1155/2022/5909033

Research Article

DC-BiGRU-CNN Algorithm for Irony Recognition in Chinese Social Comments

Yuanfang Dong,[1] **Yitong Zhang**,[1] **and Jun Li**[2]

[1]*School of Economics and Management, Changchun University of Science and Technology, Jilin 130022, Changchun, China*
[2]*Department of Data Science, Jilin University of Finance and Economics, Jilin 130117, Changchun, China*

Correspondence should be addressed to Jun Li; lijun@jlufe.edu.cn

Received 25 August 2021; Accepted 18 January 2022; Published 11 February 2022

Academic Editor: Venkatesan Rajinikanth

Irony recognition is an important research direction in text sentiment analysis, which contributes to discover ironic tone to judge text emotion correctly. The object of this paper is the application of deep learning to irony recognition, and the research aims to solve the problem that existing machine learning algorithms have difficulty in discriminating ironic tones. Aiming at the irony recognition of Chinese social comments, DC-BiGRU-CNN is proposed from the perspective of structural optimization rather than a text vectorization mechanism point of view, which is a dual-channel CNN combined with BiGRU, and incorporates attention mechanism and multi-granularity convolutional neural network as the main framework. This paper first briefly discusses the difficulties of irony recognition and provides an overview of existing text sentiment analysis algorithms. This is followed by a detailed discussion of DC-BiGRU-CNN. Furtherly, it is compared with the main irony recognition methods on a social comment dataset containing Chinese ironic comments, and the experimental results show that DC-BiGRU-CNN can improve the accuracy of irony recognition.

1. Introduction

Ironic tone is such a linguistic phenomenon that a sentence can be considered to have completely opposite meaning if only literally. As ironic tone is not uncommon in Chinese text, irony recognition became a major difficulty in the field of text sentiment analysis. Unlike general text sentiment analysis, ironic tone recognition requires knowledge of deep semantics. Deep learning provides support for learning deeper features of text, while lexicon-based approaches and traditional machine learning methods are not well suited to this task. Some irony recognition algorithms [1, 2] based on deep learning have been proposed recently, and each of them has its own characteristics.

In this paper, a new deep learning algorithm named DC-BiGRU-CNN is proposed from the perspective of structural optimization, instead of text vectorization. Unlike other deep learning algorithms, DC-BiGRU-CNN not only combines two neural network structures, taking into account both global temporal and local feature information of the text, but also uses two channels to receive feature inputs of different dimensions, respectively. In addition, to improve the ability of finding key information, the algorithm adds an attentional feature mechanism after the convolutional neural network. Such a structure is rare in text sentiment analysis algorithms.

In order to test the performance of the algorithm and to explore the impact of the number of channels and model fusion on performance, this paper conducts comparative experiments on some of the existing deep learning methods. As the result shows, this algorithm improves the recognition ability of ironic tone.

2. Related Work

By analyzing and classifying the sentiment of large volumes of comment text, the potential valuable content can be unearthed, but in the era of big data, it is obvious that traditional manual annotation methods can no longer meet the needs of processing large volumes of data. Artificial

intelligence is the domain of computer science that focuses on the development of machines that operate like humans [3]. And nowadays, artificial neural network, which has a wide variety of practice for the solution of optimization problems in the area of data classification [4], provides powerful support for text sentiment analysis.

Irony recognition is a difficult area of text sentiment analysis. Due to depending on multiple factors such as context, background, language conventions, and sentence structure, the irony recognition needs extracting deep semantic information that deep learning methods are suitable to do actually. And the methods for irony recognition are summarized from three aspects including text feature processing, sentiment analysis, and irony recognition in this paper.

2.1. Methods of Text Feature Processing. Taking the view of text feature processing, irony recognition algorithms can be classified into the following three categories: (1) lexicon-based methods. The lexicon-based methods, such as PBLGA [5] that generates lexicon based on semantic parsing, and the method based on the frequency of exclamations, rely on collecting external information and treat the text as an unordered group of words, which makes it difficult to grasp the text semantics itself and improve the recognition accuracy. (2) Traditional machine learning-based methods: support vector machine (SVM), logistic regression (LR) [6], naive Bayes (NB), and decision tree (DT) methods [7] were used for irony recognition. Even though these methods take the overall structure of the text into account and make use of the text vector matrix, they do not have the ability to extract deeper and more abstract text feature, so they cannot work well in irony recognition. (3) Deep learning-based methods: CNN (convolutional neural network) [1], LSTM (long short-term memory) [2], and the attention mechanism can do better on modeling the text, capturing deeper text information, and understanding the semantics in depth. Therefore, they are widely used deep learning methods for irony recognition and more suitable for large-scale data [8].

2.2. Deep Learning Methods of Sentiment Analysis. Deep learning methods for sentiment analysis can be roughly divided into single-channel and multichannel algorithms according to their structure.

Among the single-channel algorithms, the algorithms that incorporate multiple neural networks are more effective than the algorithms that use a single network structure (such as LSTM [9] or the multigranularity CNN [10]). Zhang et al. [11] firstly use LSTM to extract text feature vectors, then feed these vectors into a CNN for further feature abstraction, and finally classify the text through a softmax classifier. Miao et al. [12] made use of BiGRU and GRU layers jointly after the convolutional layer. Son et al. [13] use BiLSTM to extract features from the text matrix and use soft attention mechanism based on softmax function to calculate the corresponding score and then convolute the weighted vectors.

Among multichannel algorithms, the channels can be distinguished by different criteria. The dual channel could consist of dynamic word vectors and static word vectors according to whether the word vectors can be fine-tunable [10] or be composed of word vectors and character vectors [14]. The output of the dual channel forms the mixture vector from a CNN structure.

2.3. Main Methods for Irony Recognition. Most irony recognition algorithms based on deep learning came from sentiment analysis algorithms and optimized in terms of either text vectorization mechanism or algorithm structure.

The main text vectorization methods include one-hot encoding, TF-IDF, hash encoding, GloVe, and Word2vec. Among them, the Word2vec algorithm can combine contextual information well to obtain the vector representation containing semantic information. Tay et al. [15] further proposed a two-channel algorithm, where channel 1 uses a single-layer LSTM to encode sentences and channel 2 consists of a word embedding layer and an attentional feature extraction layer based on softmax function to obtain intrasentence relations. Kolchinski et al. [16] suggested to stitch text vectors and user feature vectors, and then put into a single-layer BiGRU for feature processing. After this flow, a sigmoid classifier is used to discriminate ironic tone.

Multichannel structural optimization methods can be effective in improving performance. Sun et al. [17] proposed a dual-channel algorithm, in which the convolutional channel receives word vectors as feature extractor of semantic information, while the LSTM channel receives word order vectors as mapper of contextual semantic structure. This algorithm outperforms the traditional single network algorithm. Ji et al. [18] also proposed a dual-channel fusion model, where the two channels receive the same text vector and use LSTM and CNN parallel structures, without attentional feature mechanisms.

In addition, in terms of the output of the algorithm, irony recognition algorithms also can be divided into discrete category algorithms and continuous dimension algorithms. The discrete category algorithms have the category label "whether the text is ironic," while the continuous dimensional algorithms measure the emotional intensity of text by the feature values on different dimensions such as the positive and negative degree of emotion (valence), the degree of emotional excitement (arousal), and the degree of irony (irony) to obtain a more fine-grained expression of the emotion of text. Jin et al. [19] constructed dimensional feature representations of texts in valence-arousal two-dimensional space. Although the continuous dimensional algorithms are more fine-grained, the discrete category recognition algorithm is easier to implement, receive more intuitive results, and satisfy the needs of a variety of application scenarios.

3. DC-BiGRU-CNN Algorithm for Irony Recognition

For optimizing the discrete category algorithm, we propose DC-BiGRU-CNN, and a dual-channel algorithm incorporates BiGRU (bidirectional gated recurrent unit), which

adopts the advantageous modules of other algorithms according to the characteristics of Chinese ironic text. The structure of DC-BiGRU-CNN is shown in Figure 1.

DC-BiGRU-CNN refers to the convolutional neural network algorithm proposed by Kim [10], which uses the Word2vec model to vectorize the text and uses a multi-grained convolutional kernel in the convolutional layer. Similar to the dual-channel algorithm DCCNN proposed by Li et al. [14], DC-BiGRU-CNN is a dual-channel algorithm, with one channel receiving word vectors and the other receiving character vectors. However, each channel of DC-BiGRU-CNN uses a fusion algorithm of convolutional neural network and recurrent neural network, referring to the LSTM-CNN proposed by Zhang et al. [11]. Each channel is similar to the algorithm proposed by Miao et al. [12], which combines CNN and BiGRU, but in a different order, with the CNN placed after the BiGRU. This algorithm refers to the CBAM attentional feature module and applies the attentional mechanism after the convolutional layer.

The DC-BiGRU-CNN algorithm uses the same network structure for both channels, and each channel consists of following three main components.

3.1. Input and Embedding Layer. After preprocessing the dataset with data cleaning and word segmentation in the input layer, the punctuation marks were removed by regular expressions, and deactivated words were removed using the stop word table (created by Harbin Institute of Technology) in order to obtain word-level and character-level corpus, respectively.

The embedding layers take word-level and character-level corpus as input, respectively, and obtain two pretrained Word2vec models (as embedding models) based on the skip-gram algorithm, which are used to generate word vectors and character vectors, and then generate an initial feature vector matrix.

The Word2vec algorithm adopts a neural network with a single implicit layer, as shown in Figure 2. The variable V represents the number of words in the corpus, and N is the preset size of word vector. The input layer receives V-dimensional one-hot encoding of the basic units of a text (word or character). The hidden layer multiplies the input encoding and the weight matrix ($V \times N$) to obtain N-dimensional word embedding vectors. The values in each dimension of the V-dimensional output vector represent the predicted probability of each word from the corpus.

The text vector matrix is created by stitching the word or character vectors together by row. Suppose that a sentence consists of n words, i represents the sequence number of the word, the ith word vectors can be marked as X_i, $\oplus$ denotes the concatenation operator, then this sentence can be described as

$$X_{1:n} = X_1 \oplus X_2 \oplus \cdots \oplus X_n. \tag{1}$$

3.2. BiGRU Layer and Convolutional Layer. The former extracts global information and sequential features of a text, and the latter takes convolutional kernels of three different sizes to extract local information and spatial features of the text. Considering that placing the recurrent neural network structure closer to the original text matrix may preserve more sequential features of the text, the BiGRU layer is placed in front of the convolutional layer.

3.2.1. BiGRU Layer. GRU was proposed on the basis of LSTM (long short-term memory network). LSTM is a variant of recurrent neural network and can solve the problems of short-term memory and gradient disappearance existing in recurrent neural networks, so it is suitable for processing and predicting important events with long time sequence.

The BiGRU layer uses a bidirectional GRU structure, as shown in Figure 3, to extract sequential information from the text, and has the ability to anticipate what follows. The characteristics of the recurrent structure dictate that this layer needs higher time cost, while the time cost of BiGRU is smaller than that of the bidirectional LSTM algorithms.

Every cell in the BiGRU layer (shown in Figure 4) contains two gates, an update gate and a reset gate. x_t denotes the input vector at the t^{th} time step, which also is the output of the previous layer; h_t denotes the final memory at the current time step, which is also the output of $Cell_t$ at the t^{th} time step and is also the input of neighbor cells in the same layer; $\widetilde{h}_t$ denotes the current memory content; z_t denotes the operation result of the update gate; r_t denotes the operation result of the reset gate. Let W, W_t, and W_r denote different weight matrixes, and σ represents the sigmoid function, then the formulas of GRU are shown as follows:

$$z_t = \sigma\left(W_z \cdot \left[h_{t-1}, x_t\right]\right), \tag{2}$$

$$r_t = \sigma\left(W_r \cdot \left[h_{t-1}, x_t\right]\right), \tag{3}$$

$$\widetilde{h}_t = \tan h\left(W \cdot \left[r_t * h_{t-1}, x_t\right]\right), \tag{4}$$

$$h_t = \left(1 - z_t\right) * h_{t-1} + z_t * \widetilde{h}_t. \tag{5}$$

(2) and (3) represent the method of obtaining the gating signal for the update and reset gates, respectively, both of which are weighted by the output of the previous time step and the output of the previous layer, and both of which use the sigmoid function to map the data to a gating signal in the range [0, 1]. The closer the gating signal is to 1, the more it is "remembered," while the closer it is to 0, the more it is "forgotten." Equation (4) combines the gating signal of reset gate to calculate the memory state of the current time step, using the tanh function to map the state information to the range [-1, 1]. Equation (5) is used to determine the final memory at the current time step and consists of two main components: "selective forgetting" and "selective memory."

3.2.2. Convolutional Layer. The convolutional layer is the core structure of the DC-BiGRU-CNN algorithm. In order to obtain more information over the text space, this layer uses several convolutional kernels of different sizes to extract

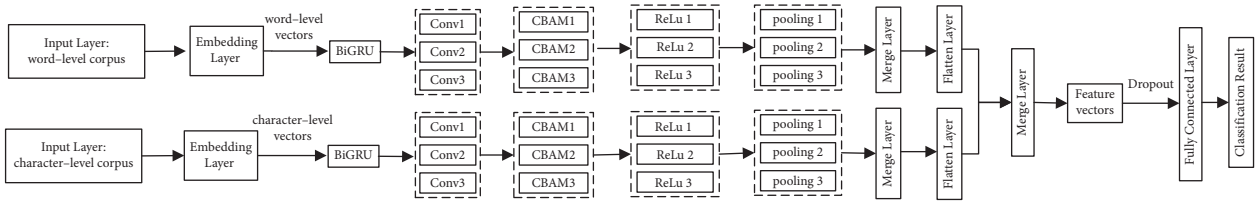

Figure 1: The structure of DC-BiGRU-CNN algorithm.

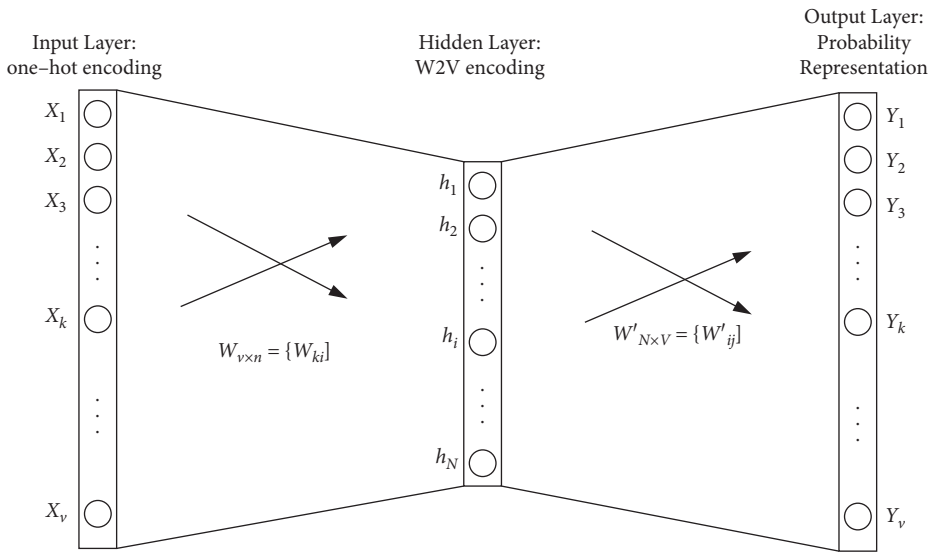

Figure 2: Word2vec model diagram.

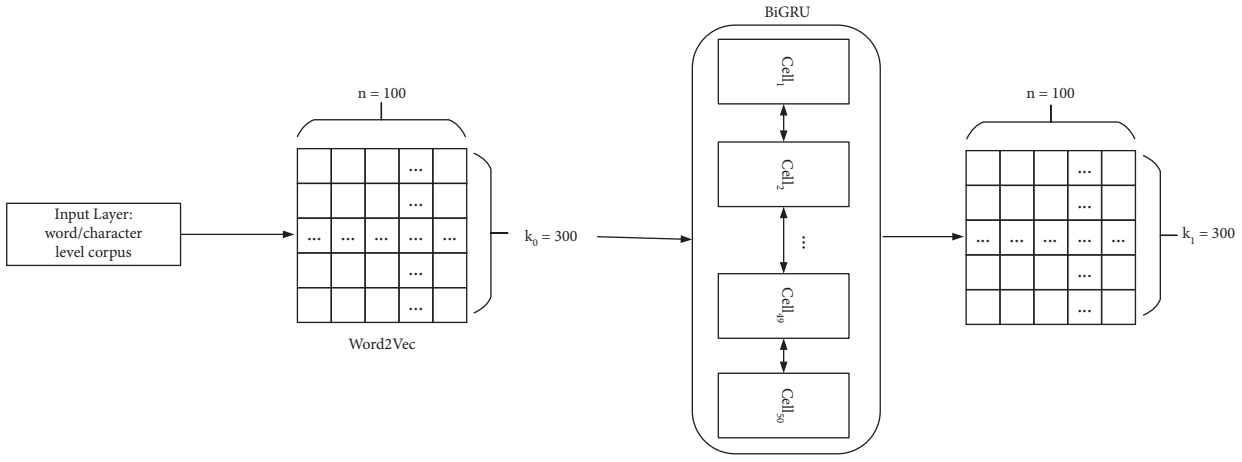

Figure 3: Schematic diagram of the BiGRU layer.

contextual information in different dimensions and performs three convolutional branches that are shown in Figure 5.

The convolutional layer is a single-layer structure with convolution step $d = 1$ and convolutional kernel sizes of 3, 5, and 7 respectively, i.e., $h = [3, 5, 7]$. Taking $h = 3$ as an example, a convolutional operation is illustrated in Figure 6. X_i represents the input feature matrix of the convolutional layer with a size of $n*K$, where n is the maximum number of words in the text, K is the dimension of the output features in the previous layer, W is the weight matrix of the convolutional kernel with a size of $h*K$, and C is the output matrix with a size of n-h+1. Let W_h ($W_h \in R^{h\times k}$) denote the weight matrix and b denote the bias term, when the convolutional kernel size is h. The convolutional eigenvalues and the feature set C can be calculated by (6) and (7), respectively:

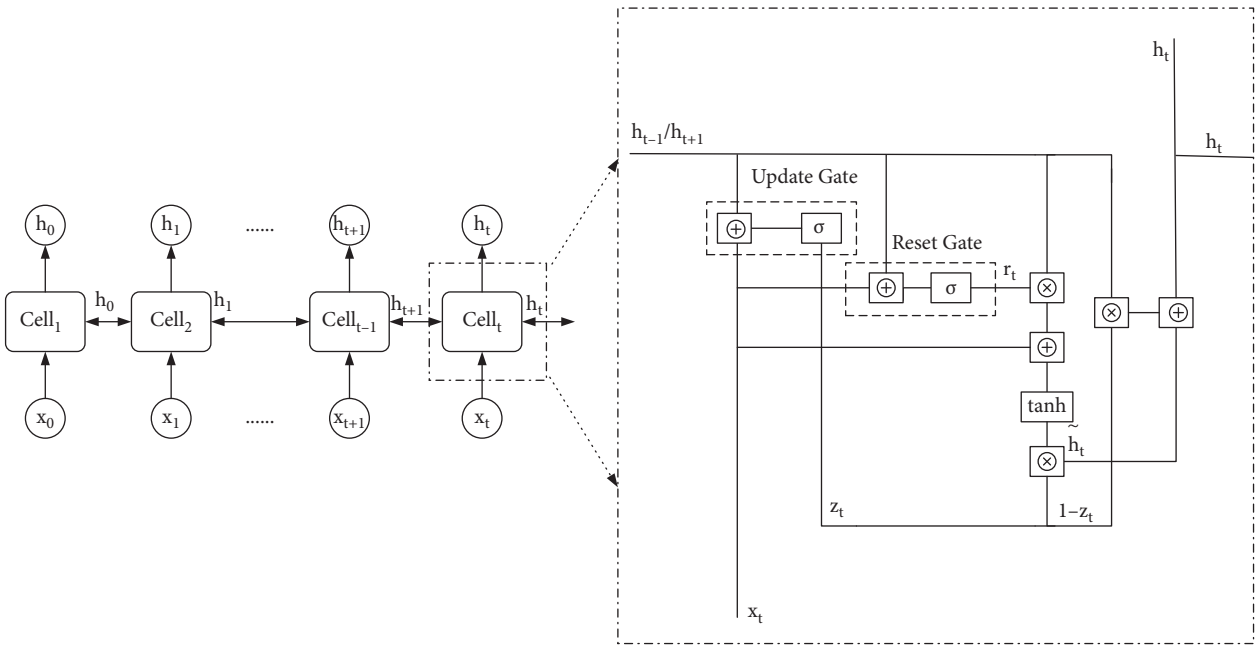

Figure 4: Structure of the BiGRU unit.

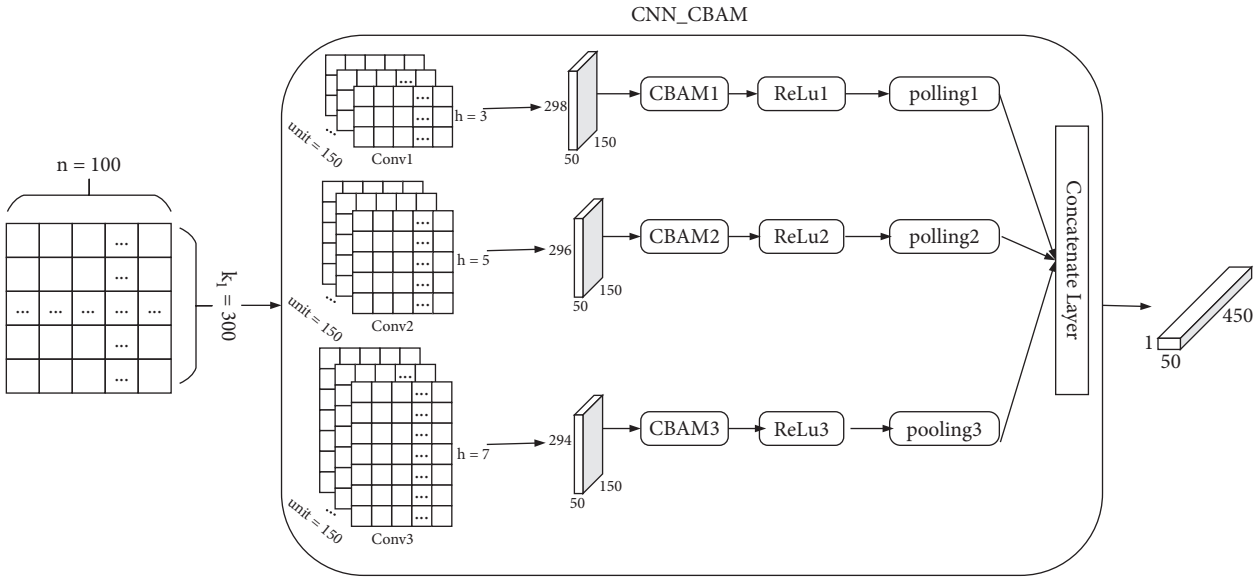

Figure 5: Schematic diagram of the convolutional layer to concatenate layer.

$$C_{h,i} = f\left(W_h X_{i:\, i+h-1} + b\right), \tag{6}$$

$$C = C_{h,1}, C_{h,2}, \ldots, C_{h,n-h+1}. \tag{7}$$

Each convolutional branch owns 150 same size kernels and will output a feature matrix with size (n-h+1)∗150.

3.3. CBAM Layer. Yin et al. [20] introduced the attention mechanism into text sentiment analysis to construct an average pooling layer, so that the algorithm can discriminate which contents are the key information in the text. And paying more attention to the key information is very important during irony recognition.

Inspired by the CBAM (convolutional block attention module) algorithm, a channel attention feature layer and a spatial attention feature layer based on convolutional operations are placed and connected after the convolutional layers, through adjusting the input and output dimensions in the network layers of the CBAM module. Taking the CBAM layer with $h = 3$ as an example, its structure diagram is shown in Figure 7.

3.4. ReLU Activation Layer, Pooling Layer, and Flatten Layer. The output of the convolution-based attention feature layer is connected to the ReLU activation function ((8) which is used to improve the nonlinear structure and accelerate the convergence.

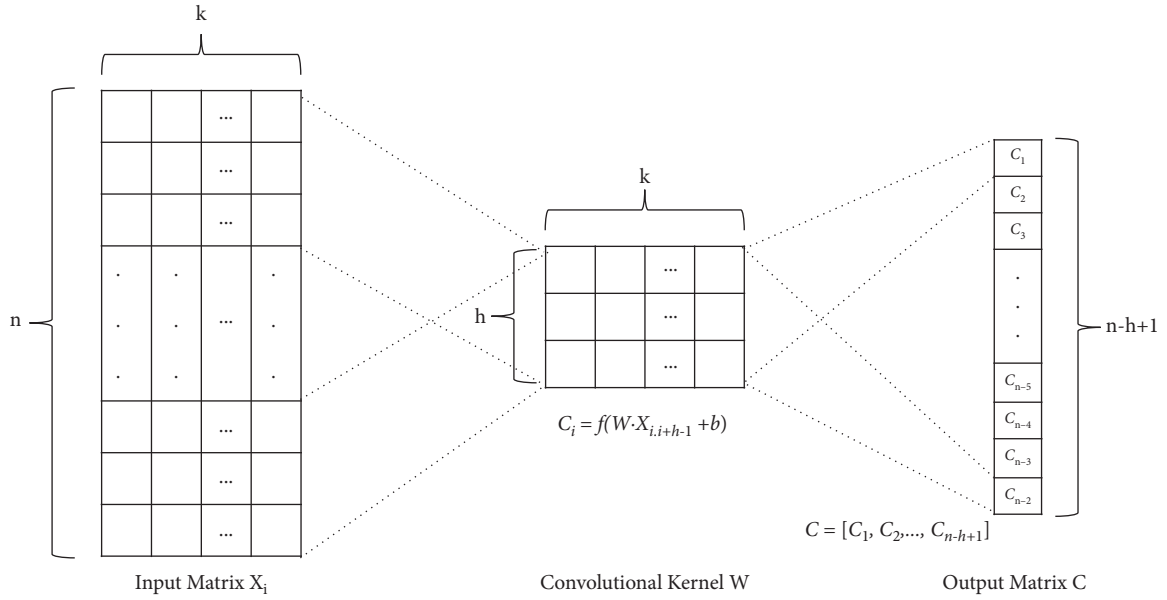

Figure 6: Schematic diagram of the convolutional operation.

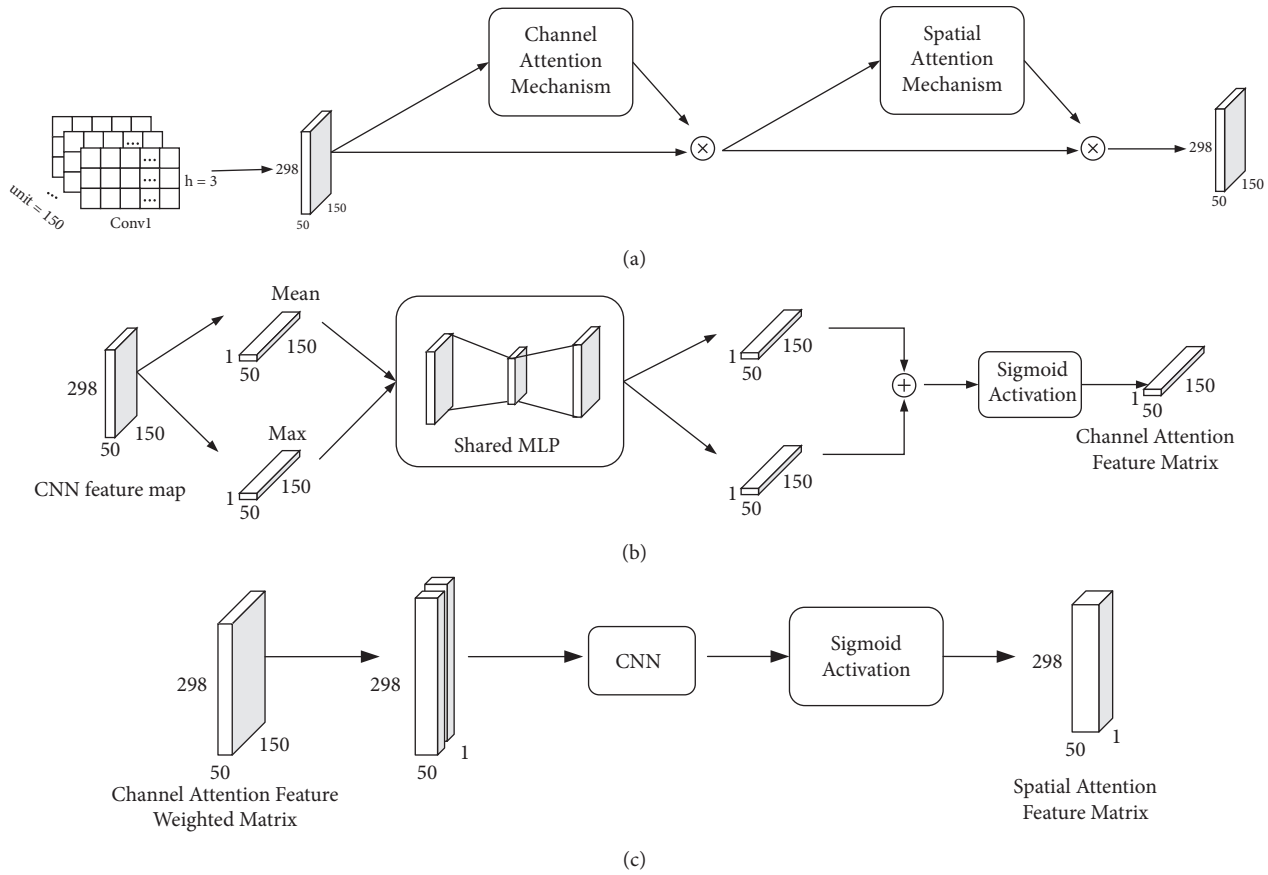

Figure 7: Structure of the CBAM layer with $h = 3$. (a) Overall structure of CBAM. (b) Structure of the channel attention mechanism. (c) Structure of the spatial attention mechanism.

$$f(x) = \max(0, x). \tag{8}$$

The Max pooling layer reduces the size of feature vector, preserves the original feature information, and is used for feature mapping to extract the optimal features from the output of the previous layer. As shown in Figure 8, followed by the pooling layer, a concatenate layer stitches the pooling results coming from different convolutional branches together and then connects a flatten layer to reduce the feature dimension of the output of the previous layer.

3.5. Concatenate Layer and Fully Connected Layer. The concatenate layer is also used to concatenate the information from the two channels, using a column-by-column stitching vector rather than averaging the vectors of the two channels in order to retain more feature information. The fully connected layer employs sigmoid function as a classifier, maps feature vectors to probabilities, and thus dichotomizes the input text for sentiment polarity.

In order to avoid overfitting, the algorithm also applies a dropout optimization strategy between the concatenate layer and fully connected layer, dropping 50% of the parameters that already have been trained in the last iteration randomly. In order to improve the nonlinear expressiveness of the algorithm, the fully connected layer uses a three-layer structure (shown in Figure 9).

3.6. Model Training and Loss Function. As the same as general neural network algorithms, DC-BiGRU-CNN algorithm uses forward propagation method to calculate the loss values of the model and uses backward propagation method to iteratively update the weight parameters in the network in the direction of gradient descent. The learning rate of the algorithm is influenced by the loss values. Using crossentropy as the loss function avoids the problem that the learning rate also decreases during gradient descent.

For the binary classification problem, the model loss function is represented by the binary crossentropy that is shown in (9). Gradient descent and the Adam optimizer were used to adjust the parameters to minimize the difference between the predicted sentiment class of the text and the true sentiment label.

$$\text{Loss} = -\sum_{x}\left[y^{(i)}\log\left(\widehat{y}^{(i)}\right) + \left(1 - y^{(i)}\right)\log\left(1 - \widehat{y}^{(i)}\right)\right]. \tag{9}$$

4. Experiments and Result Analysis

4.1. Experimental Environment. We use Jieba as the word segmentation tool, use Word2vec as the feature vector training tool, and make use of Python as the programming language. The details about related toolkits and deep learning frameworks are shown in Table 1.

4.2. Dataset Composition. We use a synthetic dataset containing 2000 comments on social platforms. In this dataset, 1000 ironic comment texts come from NTU Irony Corpus that is collected from Plurk website by Tang et al. [21], and the other 1000 nonironic comment texts are crawled from Sina microblog.

Tables 2 and 3 shows some example comments in the dataset. The label of a comment is 0 or 1 depending on whether the comment is ironic text. Most of the comments in the ironic corpus contain ironic phrases such as "very good," "great," or "could be" that are easily judged to be positive emotion if only taking the literal meaning into consideration.

To test the performance of the algorithm, 20% of the comments are selected as the test set, and the rest is then divided into a training set and a validation set in the ratio of 8 : 2. The validation set helps make a preliminary assessment of the capability of the model.

4.3. Experimental Parameter Settings

(1) The Word2vec algorithm parameters are set as shown in Table 4.

(2) Dual-channel algorithm parameters are set as shown in Table 5. They relate to the structure of the BiGRU and the convolutional layers, the number of model training iterations, and so on as shown in Tables 6 and 7, and Figure 10.

4.4. Experimental Results and Analysis. In order to verify the performance of the algorithm, a comparison experiment was conducted, and accuracy, precision, recall, and F1-value were used as evaluation metrics.

The experiments compared eight single-channel algorithms and six dual-channel algorithms. The structure comparison of those algorithms is shown in Figure 11.

The performance metrics comparison of these algorithms on the test set is shown in Table 8, the accuracy comparison of dual-channel algorithms on validation set can also be seen in Figure 12, and the iteration time of each algorithm can be found in Table 9.

During the experiments, some other algorithms, such as DCCNN-BiGRU1.0 and DCCNN-BiGRU2.0, were also constructed by adjusting the location of BiGRU and CNN, which can be seen in Figures 13 and 14. The DC-BiGRU-CNN model is finally retained because its accuracy is better than others (the performance metrics can be found in Table 10). A simplified version of the DC-BiGRU-CNN model removes the BiGRU layer from the character-level channel to make the dual-channel form a heterogeneous structure.

The experimental results show the following:

(1) Among single-channel baseline algorithms, multi-grained CNN outperforms LSTM and GRU, while fusion algorithms tend to outperform CNN baseline algorithms. The fusion algorithms can take both the extraction of textual sequential features and the local association of feature vectors into account, and have better learning ability and semantic understanding ability than the baseline algorithms on ironic sentences. To be specific, LSTM-CNN and BiLSTM-Soft-AT-CNN algorithms outperform

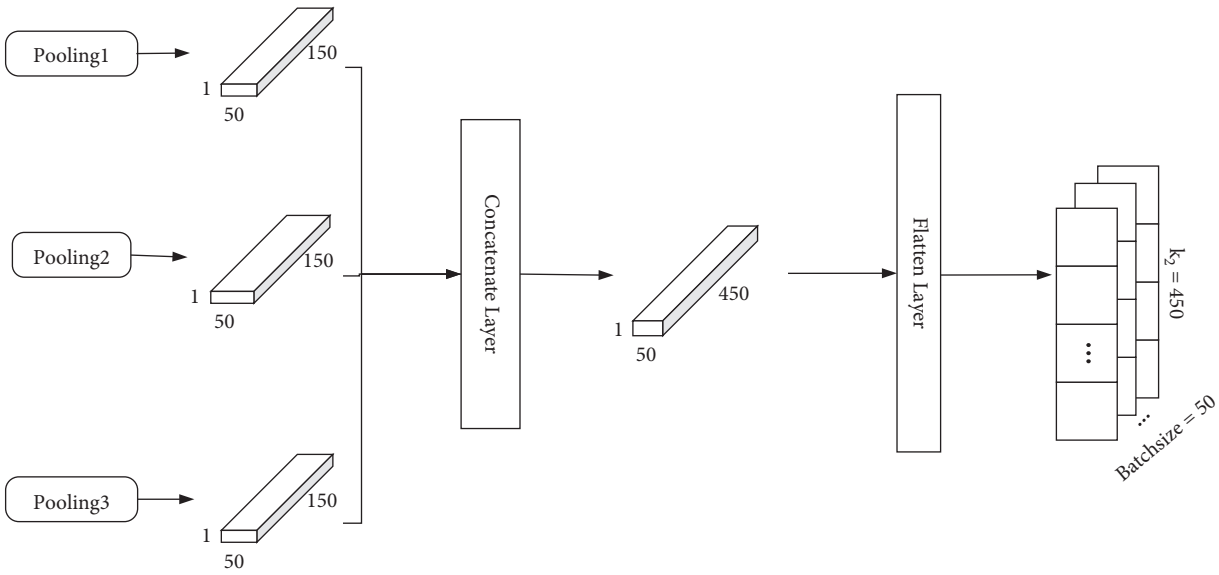

Figure 8: Schematic diagram of the flatten layer.

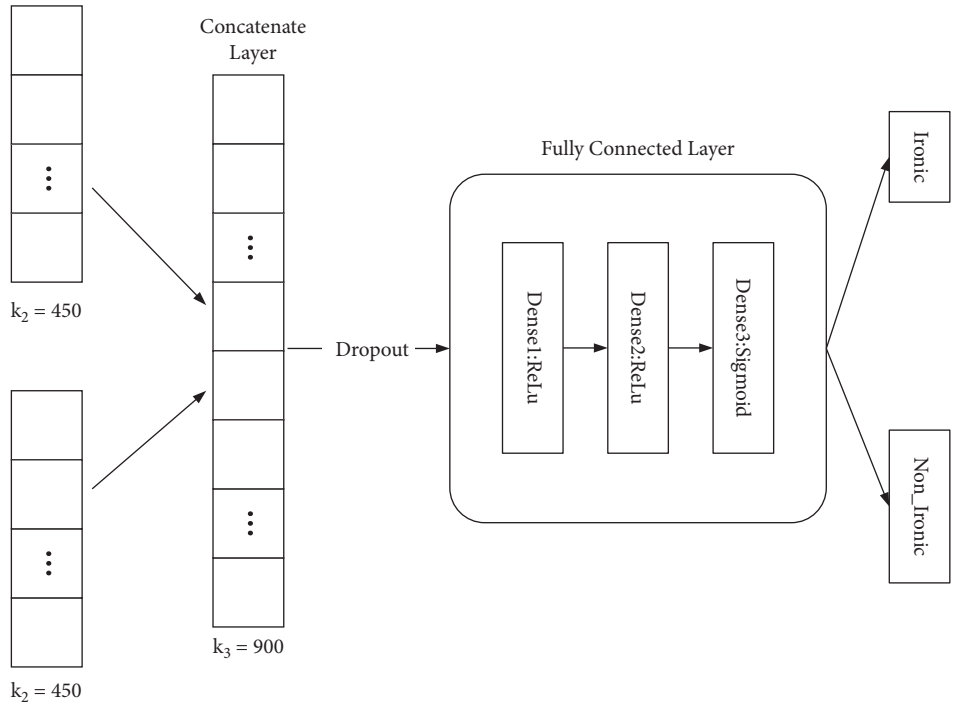

Figure 9: Schematic diagram of the concatenate layer and fully connected layer.

Table 1: Experimental environment and configuration.

Experimental environment	Environment configuration
Deep learning frameworks	TensorFlow 1.14.0
Programming environment	Anaconda 3
Programming languages	Python 3.7.3
Word separation tool	Jieba (0.39)
Word/word vector training algorithms	Word2vec (Gensim 3.8.3)

Table 2: Example comments in the synthetic dataset original Chinese version.

Label	Comment
0	机械式的可以再吵一点。
0	可以再热一点啊！难道我们现在只能待在有冷气的地方!?
1	处世低调乐观的人, 总能收获意外的惊喜。一个低调的人, 所走的人生每一步, 都是在向上登攀。低调人的生活虽不风光, 却十分惬意。恬静淡泊, 健康开朗, 乐观向上。
0	可以再倒霉一点! 最近真的是诸事不顺!
0	约早上九点的会, 到现在还没开始!可以再夸张一点!
0	好友可以再难加一点.....
1	残月疏星晚风凉, 寒窗清影徒悲秋, 一缕愁情何以消, 青樽浊酒醉天明。大家, 晚安。
1	支持本田佛山工厂的罢工斗争。
1	明天我也去跑一下。居然加表情了。刚外出遛酷小迈, 跑了一通, 通体舒泰。
0	更.....学姐啊.....你可以再瞎一点。
0	很好! 一天又要过了.....
0	很好! 我刚刚在打的一篇文章消失殆尽。
1	四级过了~在家说话都有底气了~
1	美食篇-第一次自家制糖水 - .....好好味!
0	本以为是小感冒, 但昨天去验血竟是呈疟疾阳性反应, 没想到我这么快就中标了! 非洲大陆也太特别眷顾我了吧!
1	觉得自己看起来只有12岁

Table 3: Content translation of the original Chinese version.

Label	Comment (translation version)
0	The mechanical could be a little noisier.
0	It could be a bit hotter! Are we only allowed to stay in places with cold air now!
1	People who are humble and optimistic in the world can always reap unexpected surprises. A humble person, at every step of life, is climbing upward. The life of a humble person is not wonderful, but very pleasant. Quiet and stoical, healthy and cheerful, to be optimistic.
0	Could be more unlucky! The things have been really bad lately!
0	The meeting was scheduled for 9 am and it has not started yet! It could be more exaggerated!
0	It could be harder to add friends . . ..
1	The moon is incomplete, the stars are sparse, and the wind is cool. The windows are cold and the shadows are clear, how can I kill my sorrow? Green bottle of turbid wine drunk dawn. Good night, everyone.
1	Support the strike struggle at Honda's Foshan plant.
1	I'm going to go for a run tomorrow too. And it added expressions actually! I've just been out for a walk and a good run, feeling well.
0	More... My senior... You can be a bit more blind.
0	It's good! A day is about to pass...
0	It's great that the paper I was just typing disappeared into the air.
1	I've passed CET 4 and now I have the confidence to speak at home.
1	Food—It is the first time to make homemade sugar water ... Taste good!
0	I thought it was a small cold, but yesterday I had a positive blood test for malaria. I did not think I'd fallen into the trap so soon! The African continent has been so kind to me!
1	Feeling like I just look 12 years old.

Table 4: Word2vec parameters.

Parameter	Value
Realization method	Skip-gram
Generate vector dimensions	100
Minimum of word/character frequency	3
Context window size	10
Number of parallel threads	4

Table 5: Parameters for BiGRU and convolutional layer.

Parameters	Value	Basis of parameter setting
Maximum number of words in the text (n)	300	Each comment in the dataset is short, and the maximum word count is set to 300 to suit length characteristics.
Number of BiGRU cells	50	As shown in Table 6, increasing the number of cells can preserve higher dimensional feature information, but too many cells would increase the model's computation time. A compromise is setting to 50.
Number of BiGRU layers	1	

Table 5: Continued.

Parameters	Value	Basis of parameter setting
Convolutional kernel size (h)	[3, 5, 7]	The size of the convolutional kernel determines the perceptual field of the convolutional operation. Referring to the DCCNN model proposed by Li et al. [14], convolutional kernels of sizes 3, 5, and 7 can be used to obtain richer information on contextual features.
Convolution step (stride)	1	The step size affects the accuracy of feature extraction and is set to 1 in this paper in order to consider as many lexical combination features as possible.
Number of convolutional kernels (m)	150	As shown in Table 7, increasing both the number of convolutional kernels and the number of convolutional layers can significantly improve the recognition accuracy. Considering that the fusion model already has a deeper network structure, this paper uses a three-branch structure corresponding to three different sizes of convolutional kernels, and 150 convolutional kernels for each branch.
Number of convolutional layers	3	
Dropout ratio	0.5	Using the dropout algorithm before the fully connected layer can alleviate overfitting. When the ratio of dropout is 0.5, the most network structures are generated randomly.
Dense 1 units	64	The number of neurons in the three fully connected layers is 64, 32, and 2, respectively. The number is reduced layer by layer to result in a two-dimensional vector to represent the classification probability of the model.
Dense 2 units	32	
Dense 3 units	2	
Batch_size	50	The larger the training batch size is, the more accurate the direction of gradient descent is, and the less oscillatory the intermediate results are during the training process.
Number of iterations	20	After the 20^{th} iteration of training, the accuracy for the validation set basically reached its maximum, and there is no significant improvement in the subsequent training. That can be seen in Figure 10.
Optimizers	Adam	Compared to traditional gradient descent algorithms, Adam's algorithm has higher calculation efficiency, requires less memory, and can prevent excessive learning steps by bias term.

Table 6: Performance comparison of different parameters for BiGRU

Model	Number of neurons (units)	Number of layers (layers)	Iteration time (s/epoch)	Accuracy	Precision	Recall	F1
BiGRU	25	1	9	0.7150	0.7149	0.7148	0.7148
	50	1	24	**0.7212**	0.7435	**0.7194**	**0.7145**
	75	1	35	0.7013	**0.7550**	0.6926	0.6760

Table 7: Performance comparison of convolution parameters.

Model	Number of convolutional kernels (units) per size	Number of layers by size (layers)	Iteration time (s/epoch)	Accuracy	Precision	Recall	F1
Multigrain CNN	50	1	4	0.7163	0.7415	0.7034	0.7023
	150	1	6	0.7325	0.7468	0.7327	0.7304
	50	3	10	0.7638	0.8003	0.7636	0.7533
	50	5	35	0.8150	0.8163	0.8136	0.8122
	50	7	67	0.8288	0.8596	0.8348	0.8244

multigranularity CNN algorithms which outperform the CNN algorithms using single-size convolutional kernels. In terms of time cost, CNN requires less time to train the model due to their "weight sharing" feature, while LSTM requires a longer runtime due to their chained structure and the preservation of long-term dependencies between feature vectors.

(2) The overall performance of the dual-channel algorithms is better than that of the single-channel algorithms. The performance metrics of the dual-channel algorithms receiving character vector channel are significantly higher than the other comparison algorithms.

(3) BiLSTM and BiGRU obtain better test results than LSTM and GRU. That is because they can not only extract the sequential features of the text, but also predict the next content of the current segment based on the inverse information transmission.

(4) The addition of an attention mechanism helps irony recognition and can improve the ability of model generalization. The BiLSTM-soft-AT-CNN algorithm with the soft attention mechanism focuses on the key contents in the text to help irony recognition and have the best performance among the three single-channel fusion algorithms. We also try to put forward new algorithm DCCNN-CBAM by adding the CBAM attention mechanism to the DCCNN algorithm, and it performs better than the original DCCNN. In addition, although the dual-channel algorithms always encounter a certain degree of overfitting, the algorithm with the attention mechanism gives different weights to different regions of

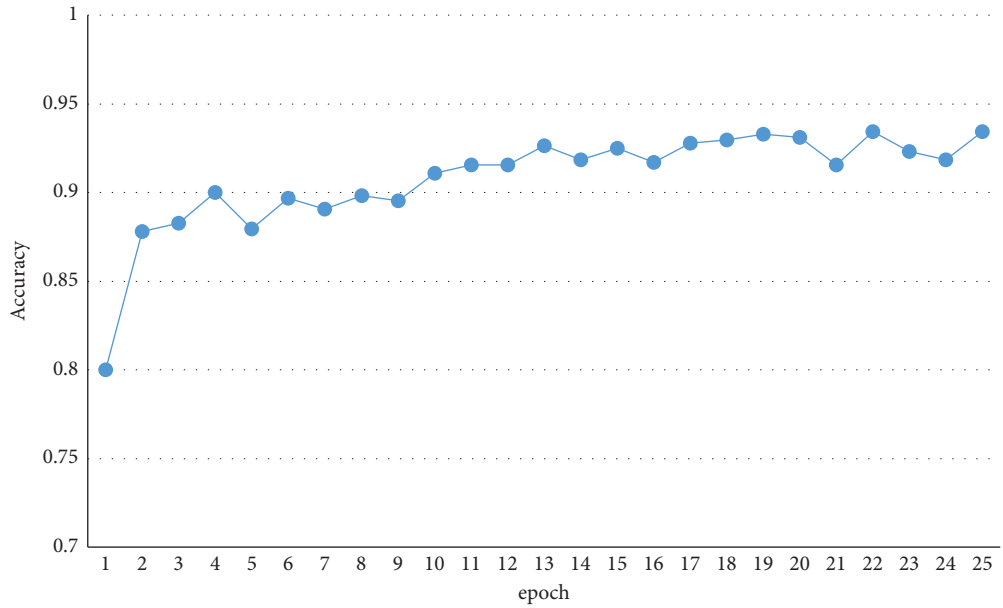

Figure 10: Model accuracy with number of iterations on the validation set.

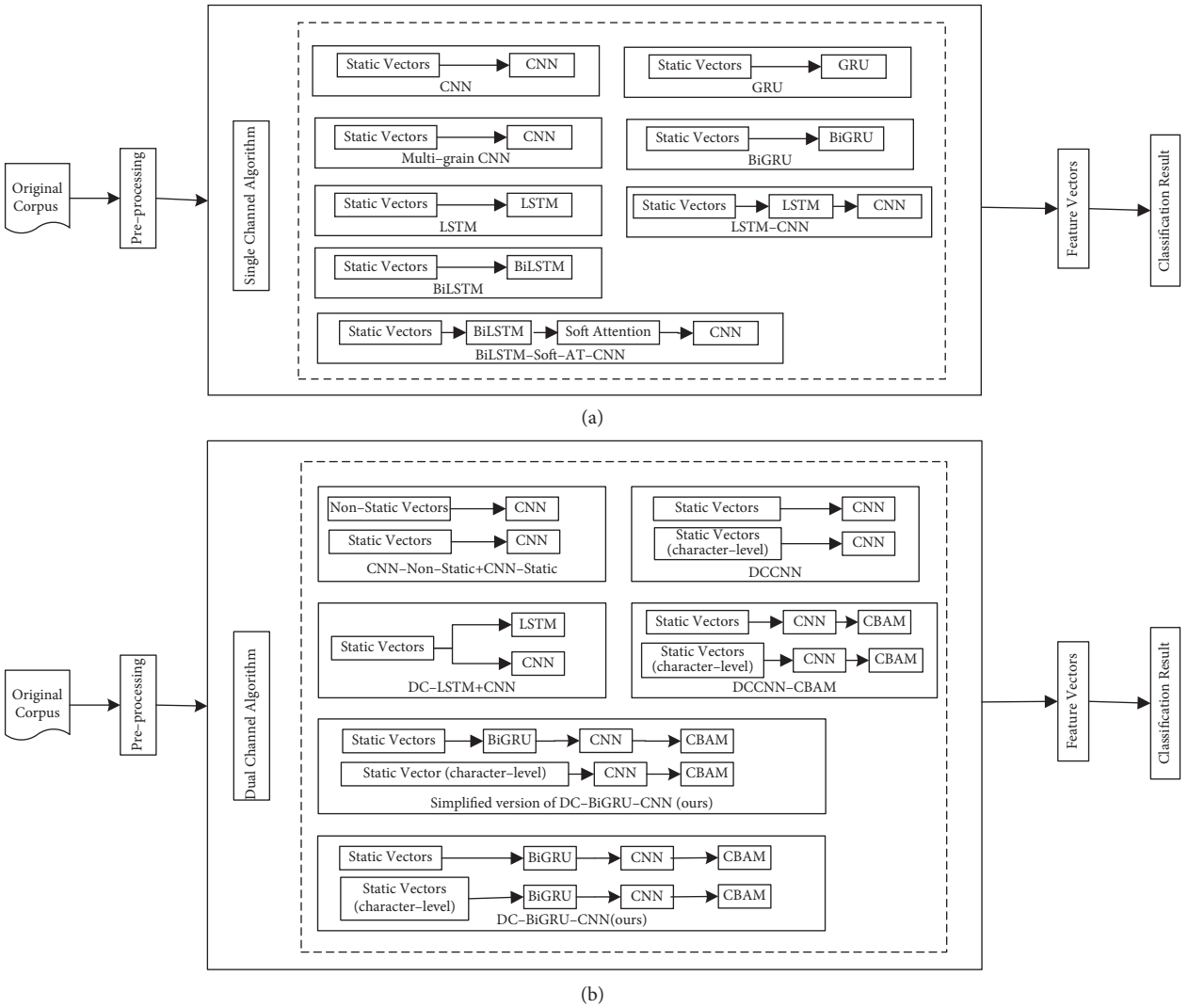

Figure 11: Comparison of algorithm structures. (a) Structure of the single-channel algorithms. (b) Structure of the dual-channel algorithms.

Table 8: Comparison of irony recognition algorithms based on deep learning.

	Algorithms	Accuracy	Precision	Recall	F1-value
Single-channel algorithm	CNN	0.6975	0.7048	0.6954	0.6933
	Multigrain CNN [10]	0.7325	0.7468	0.7327	0.7304
	LSTM [9]	0.6812	0.6890	0.6748	0.6703
	BiLSTM	0.7025	0.7026	0.7021	0.7022
	GRU	0.7025	0.7087	0.7035	0.7026
	BiGRU	0.7212	0.7435	0.7194	0.7145
	LSTM-CNN [11]	0.7563	0.7861	0.7596	0.7561
	BiLSTM-soft-AT-CNN [13]	0.8012	0.8046	0.7987	0.7987
Dual-channel algorithm	CNN-nonstatic + CNN-static [10]	0.9025	0.9026	0.9023	0.9024
	DC-LSTM + CNN [18]	0.7613	0.7582	0.7569	0.7570
	DCCNN [14]	0.9175	0.9180	0.9179	0.9175
	DCCNN-CBAM	0.9275	0.9276	0.9277	0.9275
	Simplified version of DC-BiGRU-CNN(ours)	**0.9337**	**0.9351**	**0.9353**	**0.9350**
	DC-BiGRU-CNN (ours)	**0.9362**	**0.9380**	**0.9379**	**0.9375**

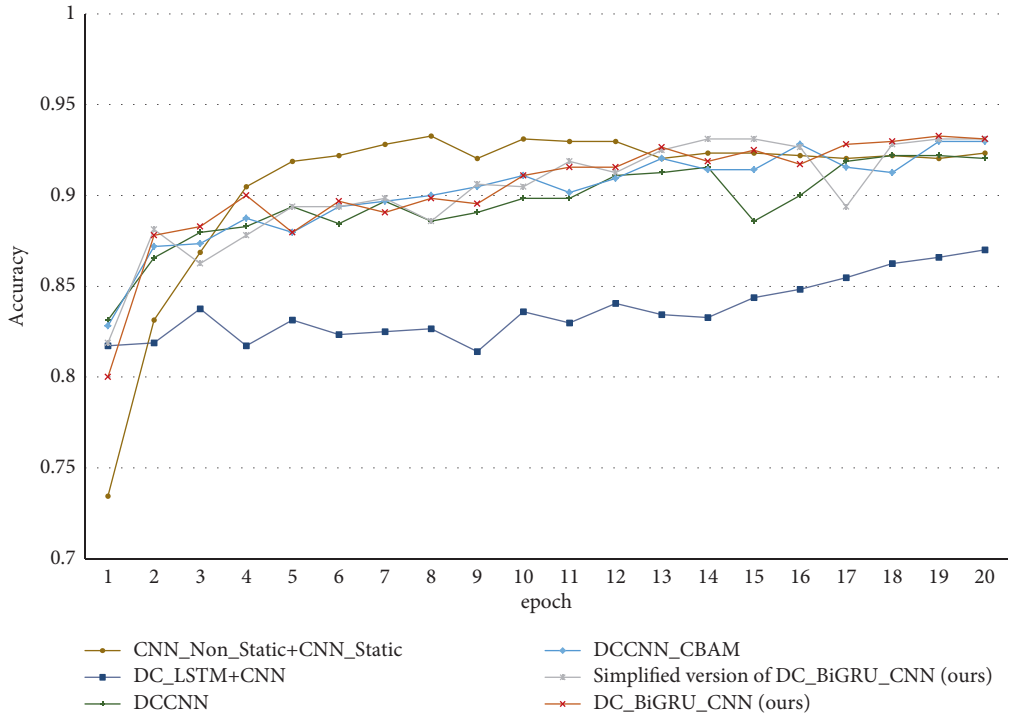

Figure 12: Comparison of the accuracy of dual-channel algorithms on validation set.

Table 9: Iteration time of the models.

	Algorithm	Iteration time (s/epoch)
Single-channel algorithms	CNN	4
	Multigrain CNN [10]	6
	LSTM [9]	14
	BiLSTM	41
	GRU	11
	BiGRU	24
	LSTM-CNN [10]	17
	BiLSTM-soft-AT-CNN [13]	72
Dual-channel algorithm	CNN-nonstatic + CNN-static [10]	23
	DC-LSTM + CNN [18]	22
	DCCNN [14]	10
	DCCNN-CBAM	24
	Simplified version of DC-BiGRU-CNN(ours)	**48**
	DC-BiGRU-CNN (ours)	**201**

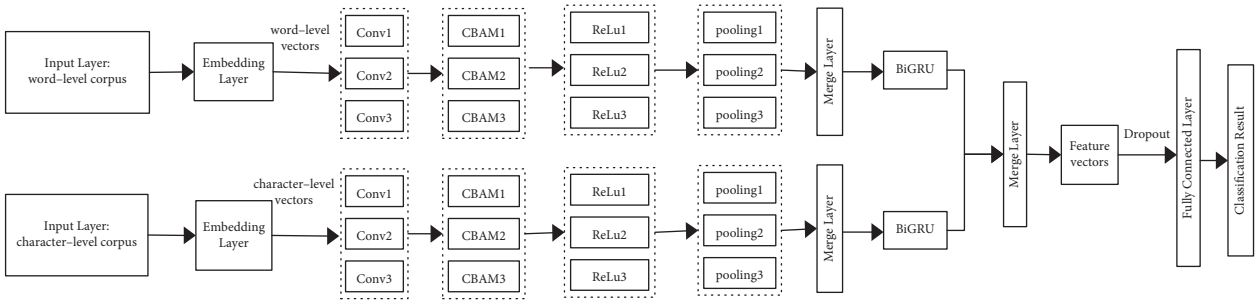

Figure 13: Structural diagram of DCCNN-BiGRU1.0 algorithm.

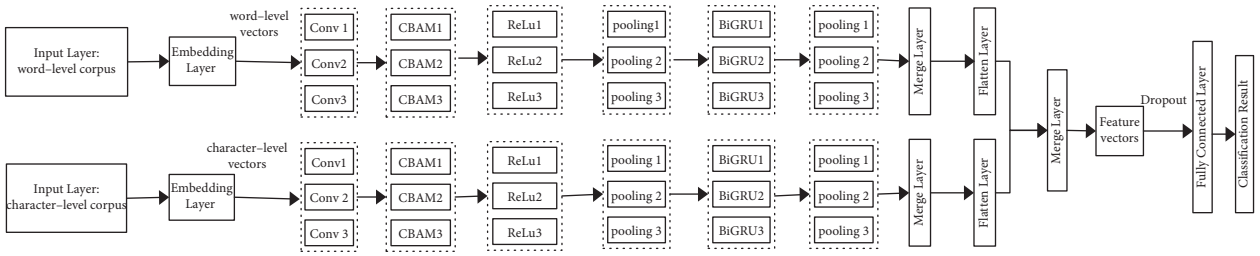

Figure 14: Structural diagram of DCCNN-BiGRU2.0 algorithm.

Table 10: Performance comparison of DC-BiGRU-CNN and DCCNN-BiGRU models.

Algorithm	Accuracy	Precision	Recall	F1
DC-BiGRU-CNN	**0.9362**	**0.9380**	**0.9379**	**0.9375**
DCCNN-BiGRU1.0	0.9325	0.9334	0.9330	0.9325
DCCNN-BiGRU2.0	0.9225	0.9225	0.9226	0.9224

the text features, so that the algorithm can capture the key information more accurately and thus improve the generalization ability of the model.

(5) The DC-BiGRU-CNN algorithm adopts dual-channel structure, combines BiGRU and CNN, and involves an attention mechanism, so it allows the model to not only capture multiple text features and have better learning ability, but also focus on the important information in the text and achieve higher accuracy. A simplified version of DC-BiGRU-CNN also achieves good test results with lower time cost.

5. Conclusion and Outlook

Irony recognition requires mining the deep semantic information that cannot rely on explicit feature extraction methods, while deep learning methods can provide support. In this paper, DC-BiGRU-CNN shows significant advantages in irony recognition. The experiments in this paper also show that the fusion model and the multichannel model have advantages in irony recognition, while the use of attentional features mechanism can also improve the model performance.

On the basis of DC-BiGRU-CNN algorithm, further new algorithms could be proposed to improve the interlayer connection by residual neural network, improve the feature selection ability of the model by combining keywords, sentiment word lexicon, lexical features, location features, etc., weight the text matrix, reduce the prediction error by ensemble algorithms, introduce dynamic text vectorization mechanism in vectorization processing layer with reference to BERT, ELMo, etc., solve the problem of multiple meanings of words, and explore the use of generative adversarial network algorithms to improve the generalization ability of the model.

Data Availability

Some or all data, models, or code generated or used during the study are available from the corresponding author by request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The research was supported by the Natural Science Foundation of Jilin Province, China (Grant no. 20150101053JC).

References

[1] Y. Lecun and Y. Bengio, "Convolutional networks for images, speech, and time series," in *Handbook of Brain Theory &*

Neural Networks, M. A. Arbib, Ed., pp. 255–258, MIT Press, Cambridge, MA, USA, 1995.

[2] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[3] S. Pervaiz, Z. Ul-Qayyum, W. H. Bangyal, L. Gao, and J. Ahmad, "A systematic literature review on particle swarm optimization techniques for medical diseases detection," *Computational and Mathematical Methods in Medicine*, vol. 2021, Article ID 5990999, 10 pages, 2021.

[4] W. H. Bangyal, J. Ahmad, and H. T. Rauf, "Optimization of neural network using improved bat algorithm for data classification," *Journal of Medical Imaging and Health Informatics*, vol. 9, no. 4, pp. 670–681, 2019.

[5] S. K. Bharti, K. S. Babuand, and S. K. Jena, "Parsing-based sarcasm sentiment recognition in twitter data," in *Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis & Mining*, pp. 1373–1380, Vancouver British Columbia Canada, August 2015.

[6] R. Gonzálezibáñez, S. Muresan, and N. Wacholder, "Identifying sarcasm in twitter: a closer look," in *Proceedings of the Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers*, pp. 581–586, Linguistics, Portland, Oregon, USA, June 2011.

[7] A. Reyes, P. Rosso, and T. Veale, "A multidimensional approach for detecting irony in twitter," *Language Resources and Evaluation*, vol. 47, no. 1, pp. 239–268, 2013.

[8] Y. Li, Z. X. Li, and L. Teng, "Comment sentiment analysis and sentiment word detection based on attention mechanism," *Computer Science*, vol. 47, no. 1, pp. 186–192, 2020.

[9] W. Xin, Y. Liu, C. Sun, B. Wang, and X. Wang, "Predicting polarities of tweets by composing word embeddings with long short-term memory," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on on Natural Language Processing*, pp. 1343–1353, Beijing, China, July 2015.

[10] Y. Kim, "Convolutional neural networks for sentence classification," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 1746–1751, Doha, Qatar, October 2014.

[11] R. Zhang, L. Honglak, and R. Dragomir, "Dependency sensitive convolutional neural networks for modeling sentences and documents," in *Proceedings of the 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1512–1521, San Diego, CA, USA, June 2016.

[12] Y. L. Miao and Y. C. Ji, "Application of CNN-BiGRU model for sentiment analysis of short Chinese texts," *Information Science*, vol. 39, no. 4, pp. 85–91, 2021.

[13] L. H. Son, A. Kumar, and S. R. SangwanArora, "Sarcasm detection using soft attention-based bidirectional long short-term memory model with convolution network," *IEEE Access*, vol. 7, Article ID 23328, 2019.

[14] P. Li, Y. M. Dai, and D. H. Wu, "Application of two-channel convolutional neural network in text sentiment analysis," *Computer Applications*, vol. 38, no. 6, pp. 1542–1546, 2018.

[15] Y. Tay, L. A. Tuan, S. C. Hui, and S. Jian, "Reasoning with sarcasm by reading in-between," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pp. 1010–1020, Melbourne, Australia, July 2018.

[16] Y. A. Kolchinski and C. Potts, "Representing social media users for sarcasm detection," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 1115–1121, Edinburgh, UK, Belgium 2018.

[17] X. Sun, J. J. He, and F. J. Ren, "A hybrid neural network model based on multi-feature fusion for sarcastic discriminative discourse," *Journal of Chinese Informatics Processing*, vol. 30, no. 6, pp. 215–223, 2016.

[18] X. F. Ji, M. D. Li, and W. G. Tao, "Twitter sentiment analysis based on multi-channel LSTM-CNN model," *Journal of Langfang Normal College (Natural Science Edition)*, vol. 19, no. 2, pp. 21–24+37, 2019.

[19] W. Jin, L. C. Yu, K. R. Lai, and X. Zhang, "Community-based weighted graph model for Valence-Arousal prediction of affective words," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 11, pp. 1957–1968, 2016.

[20] W. Yin, H. Schütze, and B. Zhou, Xiang, "ABCNN: attention-based convolutional neural network for modeling sentence pairs," *Transactions of the Association for Computational Linguistics*, vol. 4, pp. 259–272, 2016.

[21] Y. J. Tang and H. H. Chen, "Chinese irony corpus construction and ironic structure analysis," in *Proceedings of the International Conference on Computational Linguistics:Technical Papers*, pp. 1269–1278, Dublin, Ireland, August 2014.