Hindawi
Quantum Engineering
Volume 2023, Article ID 3668960, 9 pages
https://doi.org/10.1155/2023/3668960

Research Article

Multistream BertGCN for Sentiment Classification Based on Cross-Document Learning

Meng Li,[1] **Yujin Xie**,[1] **Weifeng Yang**,[2] **and Shenyu Chen**[1]

[1]*College of Mathematics and Statistic, Hebei University of Economics and Business, Shijiazhuang 050062, China*
[2]*Alibaba Group, Hangzhou, Zhejiang 311121, China*

Correspondence should be addressed to Meng Li; mli269-c@my.cityu.edu.hk

Received 6 February 2023; Revised 7 July 2023; Accepted 30 October 2023; Published 13 November 2023

Academic Editor: Shi Hai Dong

Very recently, the BERT graph convolutional network (BertGCN) model has attracted much attention from researchers due to its good text classification performance. However, just using original documents in the corpus to construct the topology of graphs for GCN-based models may lose some effective information. In this paper, we focus on sentiment classification, an important branch of text classification, and propose the multistream BERT graph convolutional network (MS-BertGCN) for sentiment classification based on cross-document learning. In the proposed method, we first combine the documents in the training set based on within-class similarity. Then, each heterogeneous graph is constructed using a group of combinations of documents for the single-stream BertGCN model. Finally, we construct multistream-BertGCN (MS-BertGCN) based on multiple heterogeneous graphs constructed from different groups of combined documents. The experimental results show that our MS-BertGCN model outperforms state-of-the-art methods on sentiment classification tasks.

1. Introduction

Sentiment analysis, also known as opinion mining, is a basic task in natural language processing, which refers to the process of analyzing, processing, and extracting subjective texts with sentiment color using natural language processing and text classification technology. Sentiment analysis is different from traditional text information processing in that traditional text information processing only focuses on the description, while sentiment analysis focuses on the emotional information embodied in the text and extracts the relevant point of view elements. With the rapid development of various network platforms, people are more and more likely to express their opinions on a certain event or a certain commodity on the Internet, and they also like to express their feelings. Therefore, there is a lot of text information on the Internet. How to process text information efficiently and accurately from these large amounts of text data and analyze the user's emotional tendency is very important. This task is sentiment classification, which is one of the core tasks of sentiment analysis, generally classifying texts with subjective sentiment. Sentiment classification is widely used in opinion mining [1], public opinion poll [2], product analysis [3], movie recommendation [4], opinion retrieval, and other fields [5], which can extract hierarchical features of text and mine sentimental tendencies of users. In recent years, sentiment classification has become a popular research topic in the field of natural language processing, which has been widely concerned by scholars and has important academic and commercial research value. The traditional sentiment classification methods are mainly based on machine learning and dictionary-based sentiment analysis. However, with the rapid development of the Internet and the rapid change of network words, the method based on sentiment dictionary needs a lot of manpower and material resources to update the sentiment dictionary, which has certain limitations. The method based on machine learning needs to rely on manual annotation of text, and it is difficult to learn deep semantic information of text. Therefore, deep learning has rapidly become the mainstream method for sentiment classification of text with its advantages of easily processing a large amount of data and strong generalization ability.

The research on deep learning used in sentiment classification began in 2006, when Hinton [6] proposed a fast learning algorithm that introduced hierarchical structure into a neural network. This method can perform feature learning well and solve the complex training problems of deep neural networks. For the study of convolutional neural networks, Kim [7] carried out experiments on English texts using the convolutional neural network (CNN) model, and the experimental results showed that the classification accuracy of the CNN model was higher, so CNN began to explore the text classification task. After the model is proposed by Zhang and Byron [8], it is also proposed to use the CNN model for text classification. Different from Kim, the sentences in this model are still arranged in the form of a sentence matrix rather than converted into vectors. Based on the combined storage concept and the theory of distributed parallel processing of the Hopfield network, Jordan and Michael [9] proposed the recurrent neural network (RNN). Recently, large-scale pretraining has demonstrated its effectiveness in a variety of NLP tasks [10]. The large-scale pretraining model is trained on a large-scale unlabeled corpus in an unsupervised manner and can learn the implicit but rich textual semantics of a language on a large scale. By combining the advantages of large-scale pretraining and inductive learning, Lin et al. [11] proposed a text classification model named BertGCN, which constructed a heterogeneous graph with word or document nodes for the corpus, initialized node embeddings with trained BERT representations, and used graph convolutional network (GCN) for classification.

Many GCN-based methods have achieved high performance in text classification. However, these models directly use the documents and words of the original data as the nodes of the graph, which may ignore some useful information in the dataset. For example, if there are large differences between the documents in a certain class, this class will not have strong discrimination compared with other classes, which may lead to difficulty in classification between different classes of the dataset. In order to better solve this problem, we propose multistream BertGCN (MS-BertGCN) classification model based on cross-document learning and apply the model to sentiment classification. Specifically, we first perform a combination of documents in the training set using within-class similarity. In each class, we calculate the similarities between different documents and combine the documents with the lowest similarity to obtain groups of combined documents. Then, we use each group of combined documents to train single-stream BertGCN. Finally, we construct our multistream BertGCN (MS-BertGCN) to obtain higher classification accuracy by fusing the classification scores of all single-stream BertGCNs. The flowchart of our method is shown in Figure 1. The main contributions are summarized as follows:

(i) We propose the MS-BertGCN model for sentiment classification based on a combination of multiple BertGCN models. To our best knowledge, this is the first research to fuse multiple BertGCN models for this task.

(ii) We propose the novel cross-document learning to train our GCN-based model.

(iii) We conducted extensive experiments on multiple sentiment datasets to demonstrate that our proposed model has better classification performance than other advanced methods.

2. Related Work

With the development of deep learning technology, deep learning models for text classification have become the mainstream solution. In the sentiment analysis method based on deep learning, researchers have proposed convolutional neural network (CNN) [8], recurrent neural network (RNN) [12], long short-term memory network (LSTM) [13], and other neural networks for better classification. As one of the important models in deep learning, CNN was first proposed by Fukushima in 1980. It is not only widely used in the field of computer vision but also widely used in the field of natural language processing (NLP). Zhang and Byron [8] successfully applied CNN to the sentiment analysis task for the first time. Tang et al. [13] proposed to use a long short-term memory network (LSTM) to model the emotional relationship between sentences, which solved the defects of gradient disappearance and gradient explosion. Huang et al. [14] proposed that syntactic knowledge can be encoded in neural networks (RNN and LSTM) and experiments showed that it was effective in improving the accuracy of sentiment text classification. Zhang et al. [15] proposed a three-way enhanced convolutional neural network model named 3W-CNN and achieved better text classification performance than CNN. Xing et al. [16] proposed a novel parameterized convolutional neural network for aspect-level sentiment classification, and experiments demonstrated that CNN-based models achieve excellent results on sentiment datasets. HEAT model further used the hierarchical attention mechanism to capture aspect information to complete sentiment analysis of specific aspects of sentences, so as to improve the accuracy of fine-grained sentiment analysis. Mohan and Yang [17] proposed a convolutional neural network text sentiment classification model based on the key sentences enhancement. Jiang et al. [18] proposed a fine-grained LSTM-CNN attention classification model, which makes good use of the ability of attention to fuse LSTM long-range dependencies and CNN local features and improves the accuracy of sentiment classification effectively. Due to the poor performance of traditional CNN models, especially for transition sentences, Zhang et al. [19] proposed a Piecewise Pooling Convolutional Neural Network (PPCNN) for sentiment classification.

2.1. Graph Neural Networks. Graph neural network (GNN) has also received a lot of attention, which is very effective for tasks with rich relational structures and can store the global structure information of a graph in a graph representation. Graph convolutional network (GCN) is an implementation

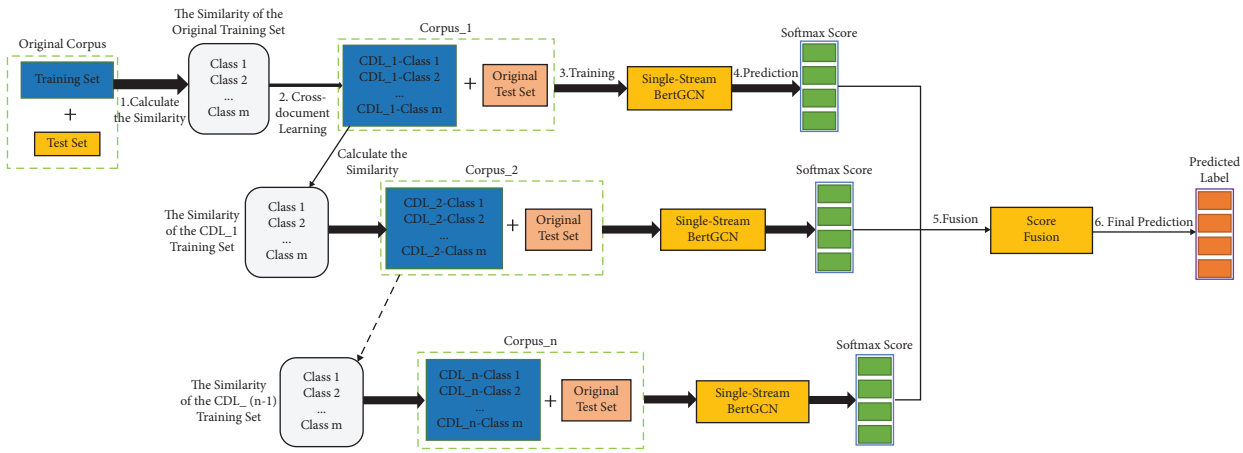

Figure 1: The schematic of our MS-BertGCN based on cross-document learning.

of graph neural network, which implements convolutional operations on topology graphs with the help of the theory of graphs. GCN proposed by Thomas and Max [20] is improved based on spectral graph convolution, using the degree matrix and adjacency matrix of graphs instead of complex eigen-decomposition operations. The author also proposes a layered linear model, which restores the expressiveness of the convolution filter function by stacking multiple convolutional layer and gets rid of the limitation of explicit parameter setting limited by Chebyshev inequality. GCN can alleviate the overfitting problem of adjacent regions structure of nodes in the classification problem by using a wider node distribution. In addition, from the perspective of operation cost, layered linear operations can further build a deeper network model.

TextGCN [21], which is proposed by applying graph convolution to text classification tasks, constructs the whole corpus into a large topological graph, with words and documents in the corpus as points in the graph and the relationships between words and documents as edges in the graph. The document-word edge is constructed by word frequency and document frequency of words; word-word edges are based on the word's global cooccurrence information. The word cooccurrence information is counted by sliding a fixed-size sliding window in the corpus, and then the weight of the connection between the two word nodes is calculated using the node mutual information (pointwise mutual information, PMI). TextGCN can capture the relationship between documents and words and the global cooccurrence information of words, and the label information of document nodes can also be passed to other words and documents through neighbor nodes. The classification experiment results show that the model can be better than some existing text representation models, such as CNN, LSTM, and Fasttext [22], as well as some models based on graph network, including SK-GCN [23], AGCN [24], graph-CNN-c [25], graph-CNN-s [26], and graph-CNN-f [27]. By improving GAT, a BiGAT model [28] was proposed to describe the contextual information of sentences. Experiments prove that BiGAT can effectively improve the speed of text classification and ensure the accuracy of text classification.

2.2. Pretraining Model. Pretrained language models (PLMs) are currently the most powerful models for natural language tasks. The model can perform unsupervised pretraining of network models using a large-scale unlabeled text corpus, and the trained network can be fine-tuned directly in various downstream NLP tasks to obtain better results. BERT is a kind of pretraining model, which uses the coding layer of the transformer network structure as the main framework of the algorithm. By proposing masked language model (MLM) and next sentence prediction (NSP), multitask training of NSP realizes the two-way propagation of data flow and solves the problems existing in the one-way language model effectively. In addition, the model takes results to a whole new level by using more powerful machines to train larger amounts of data and generate higher-quality textual representations for downstream tasks. This model was first proposed in 2018 [10] and refreshed the optimal results of 11 tasks of NLP.

RoBERTa [29] introduced the dynamic mask mechanism and deleted the next sentence prediction task in the pretraining stage. In addition, the model also increased the training data and some training parameters, such as sequence length and the number of texts per training. The SpanBERT model [30] improved BERT's MLM pretraining task. First, the model did not randomly cover a single word like BERT but randomly cover a continuous range of words. Second, the model incorporated the span boundary objective (SBO) [31]. Wang et al. [32] proposed a novel structural pretraining that extends BERT by combining word structure goals with sentence structure goals to utilize linguistic structure information in contextual representation. The model can explicitly model the language structure by forcing it to reconstruct the correct word and sentence order for prediction. In order to improve BERT's running time, a new method of knowledge distillation is proposed to compress the model, which not only saved running time and memory but also ensured strong computing power [33]. The MobileBERT model [34] and the Bert-large-like number of layers added a bottleneck mechanism to the transformer in each layer. Although the mechanism made the transformer in each layer narrower, the model did not lose its balance with the self-attention and feedforward layers. Then, a new layer-by-layer

adaptive mass optimization technique [35] was proposed, which could reduce BERT's training time from 3 days to 76 minutes. ALBERT [36] is a lightweight BERT model, which improves the traditional BERT model in two aspects: the first aspect is mainly to reduce the parameters and running time of the traditional model. The second aspect is mainly to improve the accuracy of the model in processing downstream tasks. During pretraining, ALBERT replaced the traditional next sentence prediction task with a sentence order prediction task (Sentence Order Prediction, SOP). This method not only makes it easier to generate pretraining samples but also improves the accuracy of the pretrained model in downstream NLP tasks. More recently, a more efficient pretraining task and framework [37] was proposed, which effectively combine BERT with a GAN-like structure. Unlike BERT, this pretraining task enables the model to learn all the words in the input sentence, rather than just the obscured words. Therefore, this method enables the model to learn more detailed semantic information more efficiently.

Different from the above methods, our model utilizes the fusion of multiple BertGCNs for sentiment classification based on cross-document learning which can retain more rich information from the corpus.

3. Method

In this section, we describe our proposed model in detail. Multistream BertGCN model (MS-BertGCN) is obtained by fusing multiple single-stream BertGCN models. Each single-stream BertGCN model is constructed based on the BERT module and GCN module according to the method of Lin et al. [11].

The overall algorithm is shown in Algorithm 1. Among them, lines 2–5 are the construction process of the single-stream BertGCN model.

3.1. Construction of a Graph

3.1.1. Combination of Documents. We first build a corpus according to a certain class of documents in the training set and use the TF-IDF model to process the corpus. Then, we calculate the similarities between the within-class of documents in the corpus and combine the documents with the lowest similarity. Finally, we repeat the above steps to obtain groups of combined documents.

Considering that the sentiment classification is based on the semantics of the text, the similarity measure between documents is based on the semantic similarity, and cosine similarity is used to calculate the similarity of documents. The common distances for semantic similarity measure include Euclidean distance and cosine distance. Cosine distance measures the relative difference in direction, while Euclidean distance measures numerical differences. For sentiment classification, cosine similarity is more suitable.

$$S_{\min} = \min\left(S_1, S_2, \ldots, S_t\right), \tag{1}$$

where m is the number of documents in each class and $t = \binom{m}{2}$ means that each two of the m documents are selected to calculate the similarity of documents. Based on the documents (combined documents in training and original documents in the testing set), we then construct a heterogeneous graph for the proposed model.

3.1.2. Heterogeneous Graph Construction. We need to build a heterogeneous graph composed of nodes and edges. There are two types of nodes in the graph network: documents (combined documents in training and original documents in the testing set) and words, where words are nonrepeating words in documents. The weight between words and documents is defined by TF-IDF (term frequency inverse document frequency) and the weight between words is defined by PMI [38] (positive point wise mutual information).

3.1.3. Node Feature Initialization. The input feature matrix of our model is defined as follows:

$$X \in \mathbb{R}^{\left(n_{\text{doc}}+n_{\text{word}}\right)\times d}, \tag{2}$$

where n_{doc} represents the number of documents in the graph, n_{word} represents the number of words, and d represents the dimension of the feature vector of the nodes.

In order to take advantage of the ability of the BERT to pretrain on large-scale unlabeled corpus, we initialize all document nodes X_{doc} of our GCN with BERT. For a fair comparison, we initialize all word nodes of GCN to zero instead of using a random initialization strategy, as in [11].

After obtaining the feature vectors of nodes, X is input into our GCN model based on the built heterogeneous graph to train the GCN model. The output of the i-th GCN layer is calculated as follows:

$$L^{(i)} = \rho\left(\widetilde{A}L^{(i-1)}W^{(i)}\right), \tag{3}$$

where ρ represents the activation function, $\widetilde{A}$ represents the normalized adjacency matrix, $W^{(i)} \in \mathbb{R}^{d_{i-1}\times d_i}$ represents a weight parameter, and $L^{(0)} = X$ represents the input feature of this model.

After graph propagation, the output of the last layer of GCN is used as the input of softmax:

$$Z_{\text{GCN}} = \text{softmax}\left(g\left(X, A\right)\right), \tag{4}$$

where $g(\cdot)$ is the graph model. The model is trained using the standard cross-entropy loss function.

The GCN layer operation in our proposed MS-BertGCN models has complexity $\mathcal{O}(|\mathcal{E}|FC)$, where $|\mathcal{E}|$ represents the number of edges of the graph, F represents the number of

Inactput: A dataset for sentiment classification.
Output: Sentiment label.
(1) Load the dataset;
(2) Combine the documents in the training set based on within-class similarity;
(3) Build heterogeneous graphs (including labeled data and unlabeled data and word nodes and document nodes) and initialize document nodes with BERT model;
(4) Joint training of the BERT module and GCN module;
(5) Use the trained BertGCN for inference;
(6) Repeat 2–5 to fuse the predicted result.

Algorithm 1: The Algorithm of MS-BertGCN.

convolutional kernel parameters, and C represents the dimension of each feature vector.

The general idea of the BertGCN model is to use Bert-style models (such as BERT and RoBERTa) to initialize the features of document nodes in the text graph. These features are used as inputs to the GCN. Then, the GCN will be used to iteratively update the document feature according to the graph structure, and its output is taken as the final feature of the document node and sent to the softmax classifier for prediction. Our model takes full advantage of the complementary strengths of pretraining processing and graph models. The single-stream BertGCN based on cross-document learning is shown in the red box of Figure 2.

3.1.4. Prediction of Interpolation. Since BERT and GCN process data differently and have different model sizes, directly combining them cannot lead to model convergence. In addition, a model with too large BERT cannot load all the nodes of the entire graph at one time, which hinders the training of BertGCN.

According to the method of Lin et al. [11], we add up the two document embeddings obtained from GCN and BERT acting separately on the text to get the fusion classification.

$$\begin{aligned} Z_{\mathrm{BERT}} &= \mathrm{softmax}(\mathrm{WX}), \\ Z &= \lambda Z_{\mathrm{GCN}} + (1-\lambda) Z_{\mathrm{BERT}}, \end{aligned} \tag{5}$$

when we use λ to control the trade-off between the two prediction objectives. When $\lambda = 1$, the BERT module is not updated; when $\lambda = 0$, the GCN module is not updated; when $\lambda \in (0, 1)$, both modules can be updated, and the BertGCN overall module is adjusted by adjusting λ to achieve rapid convergence of the overall module.

3.1.5. Memory Storage and Small Learning Rate. Due to the existence of BERT, BertGCN can only load one batch instead of the entire graph at a time during training, and the memory limitation prevents the full-batch method from being applied to BERT. To this end, BertGCN uses memory bank technology to solve this problem. The memory repository M saves the features of all document nodes, separates the graph nodes from each batch during training, and each batch only needs to take a small part of the node features from it.

Specifically, the memory storage mechanism is implemented as follows:

Step 1: At the beginning of each epoch, store the document node calculated by the current BERT module in the memory repository M;

Step 2: At each iteration, for the document subscript set $B = b_0, \ldots, b_n$ selected by each Batch, use the current BERT module to calculate their document features M_B, and update in M;

Step 3: The updated M is used as the input of the GCN module to calculate the loss and train the model;

Step 4: During backpropagation, only the document nodes in the B are updated, and other nodes in the M remain unchanged.

In other words, the memory storage mechanism dynamically updates a small set of document nodes with each iteration and uses this set of nodes to train the model. This avoids reading all features into BERT for calculation at one time, greatly reducing memory overhead. However, since the document nodes are updated in batches, the features input to the model will appear inconsistent in different iteration steps of an epoch. To this end, BertGCN adopts a smaller learning rate when updating the BERT module to reduce the inconsistency between features. To speed up training, BertGCN also initializes the BERT module in BertGCN with a BERT model trained on the downstream dataset before training.

3.2. Multistream Bert Graph Convolutional Network. The multistream Bert graph convolutional network (MS-BertGCN) model that we proposed combines multiple independent BertGCN based on cross-document learning and fuses the softmax scores of each graph convolutional neural network to obtain the final prediction results. For each group of combinations of documents, the softmax score R_i ($i = 1, \ldots, n$) of a test set can be obtained respectively, and our MS-BertGCN model can be obtained by

$$F = \sum_{i=1}^{n} \alpha_i R_i, \tag{6}$$

where α_i represents the weight of R_i and F represents the score of fusion. Finally, we obtain the prediction of the original documents in the test set according to the value of F. The schematic of the MS-BertGCN model is demonstrated in Figure 2.

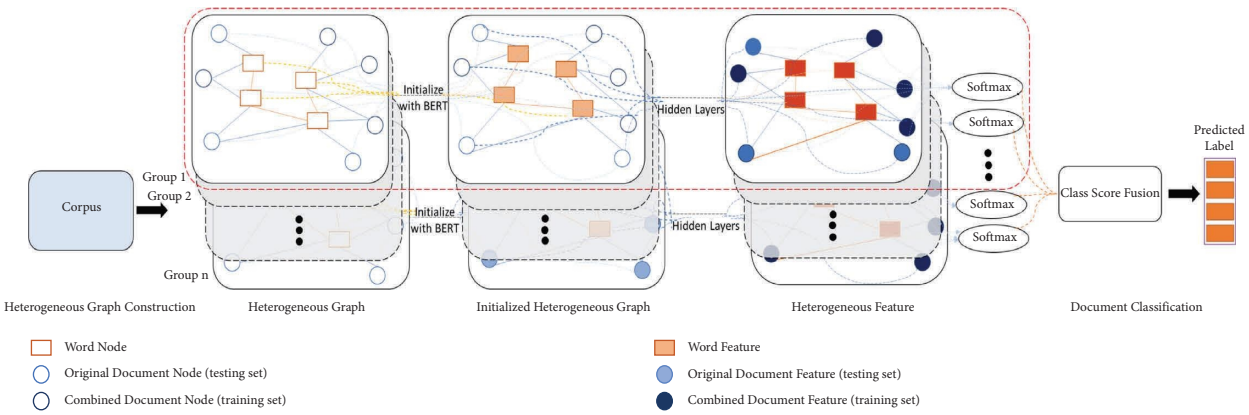

Figure 2: The schematic of our MS-BertGCN based on cross-document learning.

4. Experiments

4.1. Datasets. In this paper, three widely-used sentiment analysis datasets are applied for experiments: Movie review (MR) and the Stanford sentiment treebank (SST-2). Table 1 shows the summary statistics of these datasets.

(i) MR. The movie review dataset is a binary classification of English movie review data, with a total of 10662 samples. Each sentence in this dataset is positive and negative (denoted as 0 and 1), according to the sentiment class, there are 5331 positive statements and 5331 negative statements.

(ii) SST-2. The Stanford Sentiment Treebank (SST) is an English text sentiment classification dataset for movie reviews published by Stanford University. User reviews are divided into five levels: very negative, negative, somewhat negative, neutral, somewhat positive, positive, and very positive. After sorting on this basis (neutral comments were deleted, very positive and positive comments were marked as positive, and negative and very negative comments were marked as negative), the binary classification dataset SST-2 was obtained, which had a total of 9613 samples.

In our experiments, we use the same data preprocessing procedure and training/test splits as in the paper of Lin et al. [11]. For each dataset, we randomly sample 90% of the training set samples as the real training set, and the remaining 10% is used as the validation set.

4.2. Baseline. In our experiments, we use the baseline results of Lin et al. [11] and Yao et al. [21]. In order to prove the effectiveness of the proposed MS-BertGCN model, we compare our model with the conventional CNN model, LSTM model, and Bi-LSTM model as well as some advanced pretraining and GCN models: TextGCN, SGC, $\text{BERT}_{\text{based}}$, and RoBERTa. The details of each model are as follows:

4.2.1. CNN. The convolutional network model [7], which uses different convolution cores to convolve corpus, extract features, and finally input a pooling layer for classification, including the standard CNN for sentence classification, CNN-rand with random initialization word embeddings, and CNN-non-static with pretrained word embeddings.

Table 1: Summary statistics of datasets.

Dataset	Docs	Training	Test	Words	Classes	Avg len
MR	10662	7108	3554	18764	2	20
SST-2	9613	7792	1821	16185	2	18

4.2.2. LSTM. LSTM is a one-way LSTM network model [39, 40], which can only sequentially extract features from the corpus from front to back and use the last hidden layer vector to update parameters.

4.2.3. Bi-LSTM. This model is a bidirectional LSTM [41], which is an improvement of the traditional LSTM. It includes a forward layer and a backward layer and connects two LSTM networks with opposite timing to the same output. The forward layer can obtain the historical information of the input sequence, and the backward layer can obtain the following information about the input sequence.

4.2.4. QGRNN. QGRNN is a quantum graph recurrent neural network which is implemented using pennylane.

4.2.5. TextGCN. TextGCN [21] is a model that operates graph convolution over a word-document heterogeneous graph and can well capture local features of text.

4.2.6. SGC. Simple graph convolution [28] is a variant of GCN that reduces the complexity of GCN by removing nonlinearities and collapsing weight matrices between consecutive layers.

4.2.7. BERT. BERT [10] is a large-scale pretrained NLP model, including $\text{BERT}_{\text{based}}$ and $\text{BERT}_{\text{LARGE}}$.

4.2.8. RoBERTa. A robustly optimized BERT model [30] that improves upon BERT with different pretraining methods.

4.3. Settings. In our experiments, we trained three single-stream BertGCNs based on the three groups of combinations of documents and then constructed a three-stream BertGCN sentiment classification model. The values of the weights of the three-stream BertGCN are set as 0.02, 0.31, and 0.67 for the MR dataset and 0.76, 0.22, and 0.02 for the SST-2 dataset, respectively. These values are obtained by the grid-search method. We trained our models for at most 300 epochs and stopped the training process when the validation performance did not improve for 15 consecutive epochs. The runtime of our proposed MS-RoBertGCN model in the MR dataset and SST-2 dataset is, respectively, 132 seconds and 119 seconds. Since the model scale of BERT is too large, owing to memory restrictions, just 64 nodes of graph are loaded into each of our MS-BertGCN models in each batch of training.

In order to better compare the model performance, we designed each stream of the model according to the settings of Lin et al. [11]. For BERT and RoBERTa, we use the output features of [CLS] as document embeddings, then use the feedforward layer to derive the final prediction and use $\text{BERT}_{\text{based}}$ and two-layer GCN to implement BertGCN. For the learning rate, we initialized the GCN to $1e-3$ and the fine-tuned Bert to $1e-5$. We also implemented our model using Roberta and GAT.

4.4. Experimental Results. Table 2 shows the test accuracy of each model on the three datasets. As shown in Table 2, our proposed MS-BertGCN and MS-RobertaGCN have achieved better classification results than the baseline models in these datasets, which indicates the effectiveness of our proposed method in sentiment classification. The accuracy of using only BERT or RoBERTa is higher than that of TextGCN and SCG on MR due to the huge advantages of large-scale pretraining. On these datasets, the accuracy of BertGAT is lower than that of BertGCN because in GAT, edge weight is calculated by attention instead of TF-IDF and PPMI, which will lead to the loss of edge weight information, thus the effect is not as good as that of BertGCN.

4.5. The Effect of λ. The trade-off between GCN and BERT is controlled by λ during training, and the best value of λ may be different for different datasets. Figure 3 shows the accuracy of RoBERTaGCN under different λ. On SST-2, the accuracy always increases with the larger value of λ, due to the high performance of the graph-based method. When $\lambda = 0.7$, the model reaches the best performance.

Table 2: Test accuracy of different models.

Model	MR	SST-2
CNN	0.7498	0.8691
LSTM	0.7506	0.8920
Bi-LSTM	0.7768	0.8790
QGRNN	0.7716	0.8403
TextGCN	0.7674	0.8127
SGC	0.7591	—
BERT	0.8570	0.8869
RoBERTa	0.8943	0.8587
BertGCN	0.8600	0.8918
RoBertaGCN	0.8973	0.8833
BertGAT	0.8650	0.8910
RoBertaGAT	0.8921	0.8897
MS-BertGAT	0.8932	0.9090
MS-RoBertaGAT	0.9293	0.8984
MS-BertGCN	0.9042	**0.9293**
MS-RoBertGCN	**0.9316**	0.9085

We run all models 10 times and report the mean test accuracy. The bold values given in Table 2 represent the optimal test accuracy for different datasets.

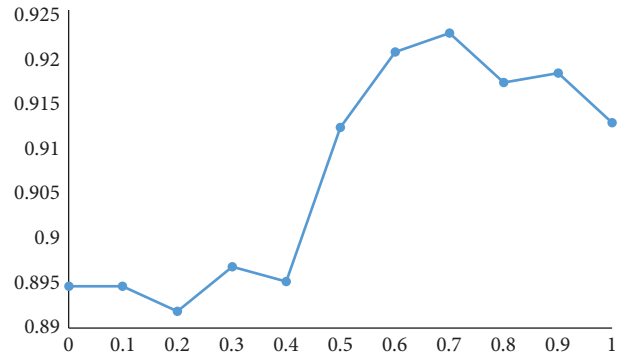

Figure 3: Accuracy of RoBERTaGCN when varying λ on the SST-2 development set.

5. Conclusion and Future Work

In this work, we propose the MS-BertGCN sentiment classification model based on cross-document learning. Firstly, we combine the documents in the training set based on similarity. Then, we group the combined documents to train BertGCN models. Finally, we fuse these BertGCN models to construct multistream BertGCN (MS-BertGCN) based on cross-document learning. The experimental results show that our proposed model can achieve the state-of-the-art performance on sentiment classification task. Considering our MS-BertGCN models just adopt a mini-batch gradient descent approach for training, loading larger corpora into our models may lead to lower training efficiency. To deal with the memory restrictions of our models, it would be interesting to research how to simplify the model

parameters and optimize the effect of the combination between the BERT and GCN models for this task in future work.

Data Availability

The data used to support the findings of this study are included within this article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Natural Science Foundation of Hebei Province (Grant no. F2019207118), Foundation of Hebei Educational Department (Grant nos. ZD2021319 and ZD2021043), and Hebei University of Economics and Business Foundation (Grant nos. 2019PY01 and 2020YB13).

References

[1] R. K. Bakshi, N. Kaur, K. Ravneet, and K. Gurpreet, "Opinion mining and sentiment analysis," *Journal Abbreviation*, vol. 10, pp. 142–149, 2008.

[2] B. O'Connor, R. Balasubramanyan, B. R. Routledge, and N. A. Smith, "From tweets to polls: linking text sentiment to public opinion time series," in *Proceedings of 4th International AAAI Conference on Weblogs and Social Media (ICWSM)*, p. 559, Washington DC, USA, May 2010.

[3] Y. Zhang, "Incorporating phrase-level sentiment analysis on textual reviews for personalized recommendation," in *Proceedings of the 8th ACM International Conference on Web Search and Data Mining*, pp. 435–440, Shanghai, China, February 2015.

[4] A. B. Goldberg and X. Zhu, "Seeing stars when there aren't many stars: graph-based semi-supervised learning for sentiment categorization," in *Proceedings of 1st Workshop on Graph Based Methods for Natural Language Processing*, pp. 45–52, New York City, NY, USA, June 2006.

[5] L. Guo and X. Wan, "Exploiting syntactic and semantic relationships between terms for opinion retrieval," *Journal of the American Society for Information Science and Technology*, vol. 63, no. 11, pp. 2269–2282, 2012.

[6] G. E. Hinton, S. Osindero, and Y. W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[7] Y. Kim, "Convolutional neural networks for sentence classification," 2014, https://arxiv.org/abs/1408.5882.

[8] Y. Zhang and C. W. Byron, "A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification," 2015, https://arxiv.org/abs/1510.03820.

[9] Jordan and I. Michael, "Serial order: a parallel distributed processing approach," *Advances in Psychology*, vol. 121, pp. 471–495, 1997.

[10] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "Bert: pre-training of deep bidirec-tional transformers for language understanding," 2018, https://arxiv.org/abs/1810.04805.

[11] Y. Lin, Y. Meng, X. Sun et al., "BertGCN: transductive text classification by combining GCN and BERT," 2021, https://arxiv.org/abs/2105.05727.

[12] A. Graves, "Generating sequences with recurrent neural networks," 2013, https://arxiv.org/abs/1308.0850.

[13] D. Tang, B. Qin, and T. Liu, "Document modeling with gated recurrent neural network for sentiment classification," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1422–1432, Lisbon, Portugal, September 2015.

[14] M. Huang, Q. Qian, and X. Zhu, "Encoding syntactic knowledge in neural networks for sentiment classification," *ACM Transactions on Information Systems*, vol. 35, no. 3, pp. 1–27, 2017.

[15] Y. Zhang, Z. Zhang, D. Miao, and J. Wang, "Three-way enhanced convolutional neural networks for sentence-level sentiment classification," *Information Sciences*, vol. 477, pp. 55–64, 2019.

[16] Y. Xing, C. Xiao, Y. Wu, and Z. Ding, "A convolutional neural network for aspect-level sentiment classification," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 33, no. 14, Article ID 1959046, 2019.

[17] Z. Mohan and X. Yang, "A key sentences based convolution neural network for text sentiment classification," *Journal of Physics: Conference Series*, vol. 1229, no. 1, Article ID 012062, 2019.

[18] M. Jiang, W. Zhang, M. Zhang, J. Wu, and T. Wen, "An LSTM-CNN attention approach for aspect-level sentiment classification," *Journal of Computational Methods in Science and Engineering*, vol. 19, no. 4, pp. 859–868, 2019.

[19] Y. Zhang, Q. Wang, Y. Li, and X. Wu, "Sentiment classification based on piecewise pooling convolutional neural network," *Computers, Materials and Continua*, vol. 56, no. 2, pp. 285–297, 2018.

[20] N. K. Thomas and W. Max, "Semi-supervised classification with graph convolutional networks," in *Proceedings of the 5th International Conference on Learning Representations*, Toulon, France, April 2017.

[21] L. Yao, C. Mao, and Y. Luo, "Graph convolutional networks for text classification," in *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, pp. 7370–7377, Hawaii, HI, USA, February 2019.

[22] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of tricks for efficient text classification," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 427–431, Valencia, Spain, April 2017.

[23] J. Zhou, J. X. Huang, Q. V. Hu, and L. He, "SK-GCN: modeling syntax and knowledge via graph convolutional network for aspect-level sentiment classification," *Knowledge-Based Systems*, vol. 205, no. 3, Article ID 106292, 2020.

[24] M. Zhao, J. Yang, J. Zhang, and S. Wang, "Aggregated graph convolutional networks for aspect-based sentiment classification," *Information Sciences*, vol. 600, pp. 73–93, 2022.

[25] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," *Advances in Neural Information Processing Systems*, vol. 29, 2016.

[26] J. Bruna, W. Zaremba, A. Szlam, and Y. Lecun, "Spectral networks and locally connected networks on graphs," 2013, https://arxiv.org/abs/1312.6203.

[27] M. Henaff, J. Bruna, and Y. Lecun, "Deep convolutional networks on graph-structured data," 2015, https://arxiv.org/abs/1506.05163.

[28] Y. Shan, C. Che, X. Wei, X. Wang, Y. Zhu, and B. Jin, "Bi-graph attention network for aspect category sentiment

classification," *Knowledge-Based Systems*, vol. 258, Article ID 109972, 2022.

[29] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, "Xlnet: generalized autoregressive pretraining for language understanding," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[30] Y. Liu, M. Ott, N. Goyal et al., "RoBERTa: a robustly optimized BERT pretraining approach," 2019, https://arxiv.org/abs/1907.11692.

[31] M. Joshi, D. Chen, Y. Liu, D. S. Weld, L. Zettlemoyer, and O. Levy, "SpanBERT: improving pre-training by representing and predicting spans," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 64–77, 2020.

[32] W. Wang, B. Bi, M. Yan et al., "StructBERT: incorporating language structures into pre-training for deep language understanding," in *Proceedings of the 8th International Conference on Learning Representations*, Addis Ababa, Ethiopia, April 2020.

[33] S. Sun, Y. Cheng, Z. Gan, and J. Liu, "Patient knowledge distillation for BERT model compression," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, Hong Kong, China, November 2019.

[34] Z. Sun, H. Yu, and X. Song, "MobileBERT: a compact task-agnostic BERT for resource-limited devices," 2020, https://arxiv.org/abs/2004.02984.

[35] Y. You, J. Li, S. Reddi et al., "Large batch optimization for deep learning: training BERT in 76 minutes," 2020, https://arxiv.org/abs/1904.00962.

[36] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "ALBERT: a lite BERT for self-supervised learning of language representations," 2020, https://arxiv.org/abs/1909.11942.

[37] K. Clark, M. T. Luong, Q. V. Le, and C. D. Manning, "ELECTRA: pre-training text encoders as discriminators rather than generators," 2020, https://arxiv.org/abs/2003.10555.

[38] K. Church and P. Hanks, "Word association norms, mutual information, and lexicography," *Computational Linguistics*, vol. 16, pp. 22–29, 1990.

[39] P. Liu, X. Qiu, and X. Huang, "Recurrent neural network for text classification with multi-task learning," 2016, https://arxiv.org/abs/1605.05101.

[40] R. Jing, "A self-attention based LSTM network for text classification," *Journal of Physics: Conference Series*, vol. 1207, Article ID 12008, 2019.

[41] K. S. Tai, R. Socher, and C. D. Manning, "Improved semantic representations from tree-structured long short-term memory networks," 2015, https://arxiv.org/abs/1503.00075.