IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 14, 2021, accepted January 14, 2022, date of publication January 25, 2022, date of current version February 2, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3146303

Graph Convolutional Network Based on Multi-Head Pooling for Short Text Classification

HONGYU ZHAO[1,2], JIAZHI XIE[1], AND HONGBIN WANG[2]
[1]Department of Economics and Management, Southwest University, Chongqing 400715, China
[2]Mashang Consumer Finance Company Ltd., Chongqing 401331, China

Corresponding author: Hongyu Zhao (asas.014@163.com)

ABSTRACT The short text, sparse features, and the lack of training data, etc. are still the key bottlenecks that restrict the successful application of traditional text classification methods. To address these problems, we propose a Multi-head-Pooling-based Graph Convolutional Network (MP-GCN) for semi-supervised short text classification, and introduce its three architectures, which focus on the node representation learning of 1-order, 1&2-order of isomorphic graphs, and 1-order of heterogeneous graphs, respectively. It only focuses on the structural information of the text graph and does not need pre-training word embedding as the initial node feature. A graph pooling based on self-attention is introduced to evaluate and select important nodes, and the multi-head method is used to provide multiple representation subspaces for pooling without adding trainable parameters. Experimental results demonstrated that, without using pre-training embedding, MP-GCN outperforms state-of-the-art models across five benchmark datasets.

INDEX TERMS Graph convolutional network, artificial intelligence, text classification, natural language processing.

I. INTRODUCTION

Text classification is a classical problem in natural language processing (NLP), which aims to assign labels or tags to textual units such as sentences, queries, paragraphs, and documents. In the past few years, scholars had proposed a series of deep learning models for that, such as models based on recurrent neural networks (RNNs), convolutional neural networks (CNNs), transformers, capsule nets, which surpass the traditional machine learning methods in various text classification tasks.

In recent years, some scholars began to study semi-supervised graph convolutional networks (GCNs) for text classification [1], [2]. The main reason is that it can be applied to many practical scenarios. Firstly, it can deal with the short text better by adding more relationships to word nodes and can be applied to the scene with sparse and ambiguous semantics and lack of context [2]. Secondly, it is also suitable for the scene with limited labeled training data, which usually leads to the poor performance of traditional neural networks [3]. As a consequence, there is a pressing need for studying semi-supervised GCNs for text classification.

The application of semi-supervised GCNs is also facing challenges. Firstly, due to different scenes, the pre-training word vector may not be able to improve the text classification effect, but increase the difficulty of graph building. Secondly, it usually builds a graph for the whole corpus so that more links or dependencies can be added to the nodes (e.g. using heuristics). Therefore, both graph building and feature extraction need to consider the calculation and memory consumption.

In this study, we present a Multi-head-Pooling-based Graph Convolutional Network (MP-GCN) for semi-supervised short text classification. MP-GCN can evaluate and select important nodes from multiple perspectives through multi-head pooling, and achieve strong classification performance with lower computational cost. Our source code is available at https://github.com/shanzhonglujie/MP-GCN. To summarize, our contributions are as follows:

(1) We propose a novel graph convolutional network (MP-GCN) for short text classification and introduce its three architectures. MP-GCN mainly focuses on the structural information of the text graph and enhances the representation

The associate editor coordinating the review of this manuscript and approving it for publication was Hiram Ponce.

learning of important nodes. Finally, it can achieve strong classification performance without combining any prior information or pre-trained embedding.

(2) Our model does not add new trainable parameters to the convolution calculation and can be applied to the feature extraction of isomorphic and heterogeneous graphs respectively. Moreover, it can output the predictive graph embeddings of words or documents to downstream tasks.

II. RELATED WORK

A. GRAPH CONVOLUTIONAL NETWORKS

Graph convolutional networks (GCNs) are mainly divided into spectral methods and spatial methods [4].

In recent years, spectral methods had received growing attention recently. Bruna *et al.* [4] first put forward the concept of GCNs in 2014, which defined graph convolution in spectrum space, with high complexity in space and time. Deferrard *et al.* [5] used k-order Chebyshev polynomials as the convolution kernel to extract features of k-order neighborhood nodes, which improved the calculation efficiency. Kipf *et al.* [1] simplified the Chebyshev network and proposed a simple, efficient graph convolutional network (GCN) with 1-order message passing.

In addition, spatial methods had also been developed rapidly. Hamilton *et al.* [6] proposed a graph sampling aggregation network (GraphSAGE), which used methods such as limiting the number of samples collected and mini-batch training to expand GCNs into inductive learning networks and solve the problem of large-scale data processing. Velikovic *et al.* [7] proposed a graph attention network (GAT), and defined aggregation function through the attention mechanism, to assign different weights to each related node and select more similar nodes for aggregation.

B. GRAPH CONVOLUTIONAL NETWORKS FOR TEXT CLASSIFICATION

The above GCNs could be applied to several tasks based on texts. Among them, text classification is an important and classical problem in natural language processing (NLP) [8]. Yao *et al.* [2] proposed Text-GCN for semi-supervised text classification. PMI and TF-IDF algorithms were used to build the topological graph containing words and document nodes, and two-layer GCN was used. Huang *et al.* [9] proposed a new graph convolutional network, which solved the problem of not supporting online testing and high memory consumption in [2], but its computational complexity is relatively high. Zhang *et al.* [10] proposed to employ GCNs on the dependency tree of sentences to capture syntactic information and lexical dependence. On this basis, they proposed a framework for sentiment classification in specific aspects. Hu *et al.* [3] built a heterogeneous graph of topic-text-entity and proposed a heterogeneous graph attention network (HGAT) based on the two-level attention mechanism for short text classification. This network could learn the importance of different adjacent nodes and the importance of different types of node information to the current node. Liu *et al.* [11] built three kinds of heterogeneous graphs to describe semantic, syntactic, and sequential contextual information and proposed a tensor graph convolutional network (TensorGCN) to harmonize and integrate heterogeneous information from three types of graphs.

In practical application, some methods either view a document or a sentence as a graph of word nodes or rely on the document citation relation to build the graph, but they don't utilize much text information [8], [12].

C. DEEP NEURAL NETWORKS FOR SHORT TEXT CLASSIFICATION

Deep neural networks, which automatically represent texts as embeddings, have been widely used for NLP tasks. Two typical deep neural networks CNNs and RNNs have shown their power for text classification [13], [14]. In the short text, the context dependence between words is usually weak, and the CNN-based model usually performs better. Considering the efficiency, the CNN-based model is more suitable for real-time application than some large models, such as Bert [15]. However, these deep learning networks cannot solve the problem of lack of training data, which prohibits them from successful practical applications.

III. PROPOSED METHOD

A. GCN

The graph convolutional networks (GCNs) are mainly used to process the data with generalized topological graph structure and explore its characteristics and laws deeply [1]. GCNs can be divided into two categories: spectral methods and spatial methods. With the deepening of the research, the spectral methods become more efficient, and their practicability becomes stronger. Among them, GCN [1] transforms spectral convolution operation in the time domain into matrix multiplication operation in the frequency domain:

$$g * x \approx \theta \left(D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \right) x, \tag{1}$$

where g is the convolution kernel in the time domain. x is the input signal. $*$ is convolution calculation. A is the adjacency matrix with self-loops, which reflects the interconnection relationship of nodes in the graph. A can be decomposed, i.e. $A = I_N + A_0$, I_N is the unit matrix, and A_0 is the adjacency matrix without self-loops. D is the degree matrix of A, and $D_{ii} = \sum_i A_{i,j}$. θ is a scalar and degenerates to 1 finally. $D^{-1/2} A D^{-1/2}$ is the normalized form of adjacency matrix A, which is used to prevent the gradient from disappearing or exploding when a multi-layer network is optimized. Equation (1) represents the convolution calculation of single layer GCN (called GCNConv), which only acts on the 1-order sub-graph of each node, i.e., in each layer, only the 1-order

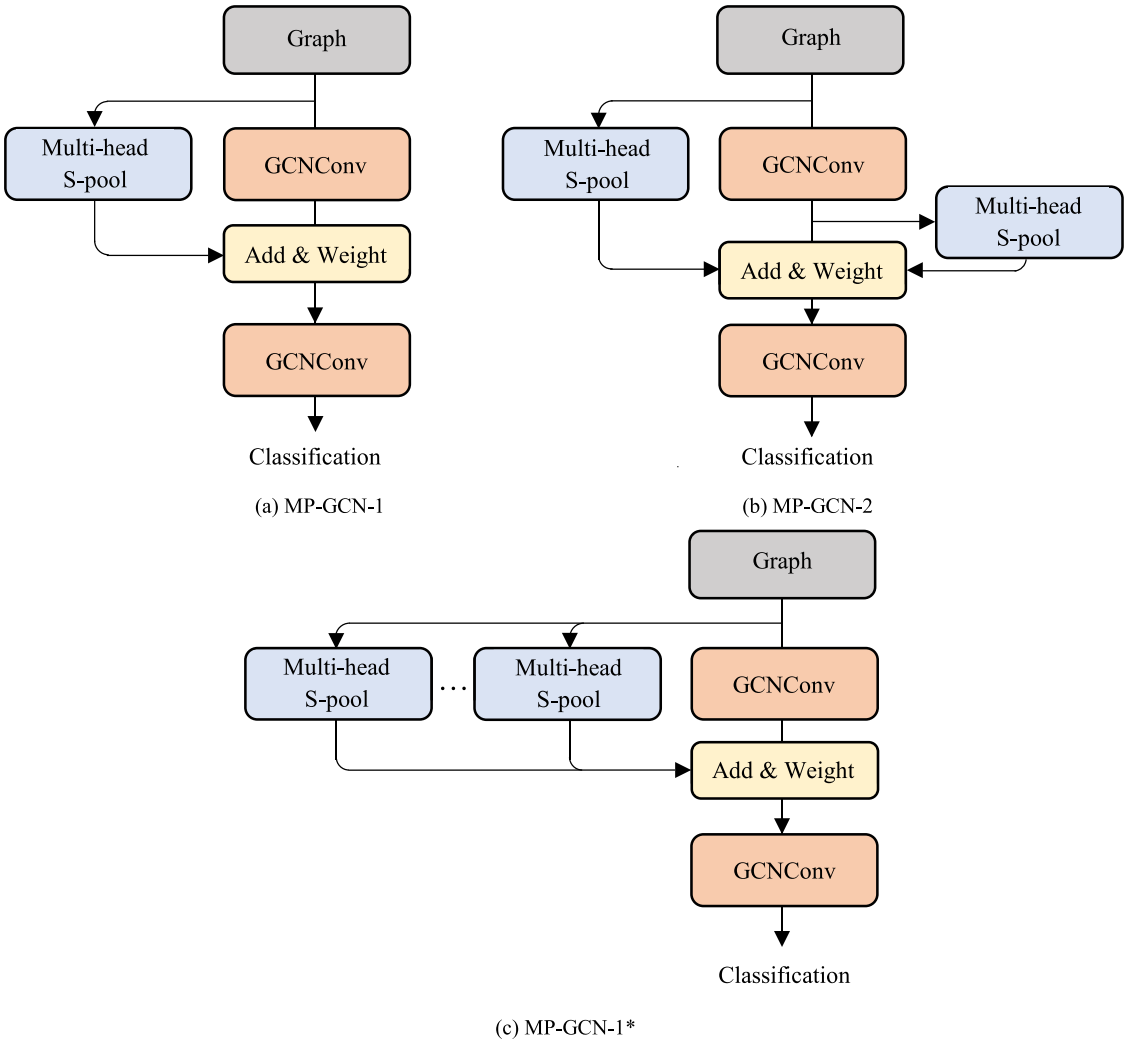


FIGURE 1. Three architectures of MP-GCN. The term *GCNConv* denotes the graph convolution layer. *Multi-head S-pool* denotes the pooling layer. *Add* represents the residual connection. *Weight* represents the weighting operation. The pooling layers in those three models have different concerns: (1) MP-GCN-1: Focusing on the 1-order nodes of isomorphic graphs. (2) MP-GCN-2: Focusing on the 1-order and 2-order nodes of isomorphic graphs. (3) MP-GCN-1*: Focusing on the 1-order nodes of heterogeneous graphs. They evaluate and select important nodes according to these concerns.

neighbors of each node are required to participate in the calculation.

After integration, the single-layer convolution formula of GCN is:

$$Z = \sigma\left(D^{-\frac{1}{2}}AD^{-\frac{1}{2}}XW\right) = \sigma\left(\tilde{A}XW\right), \tag{2}$$

where $\tilde{A} = D^{-1/2}AD^{-1/2}$, $\tilde{A} \in \mathbb{R}^{V\times V}$ is the normalized form of $A \in \mathbb{R}^{V\times V}$. σ is the activation function. $W \in \mathbb{R}^{C\times F}$ is the parameter to be trained, which is used for affine transformation of X. $X \in \mathbb{R}^{V\times C}$ is the input node feature. $Z \in \mathbb{R}^{V\times F}$ is the output. V is the number of nodes in the graph. C is the initial dimension of the node feature and F is the output dimension. Compared with the previous work, GCN reduces the number of parameters to be trained and decreases the risk of over-fitting [1]. Because GCN is mainly based on matrix multiplication, it can achieve efficient feature extraction.

B. GCN BASED ON MULTI-HEAD POOLING (MP-GCN)

The purpose of pooling methods is to reduce the size of parameters by node selection [16]–[19] (e.g. down-sampling) to generate smaller representations. However, instead of down-sampling, our pool method selects important nodes to enhance their representation learning and does not abandon non-important nodes.

According to the different combinations of the pooling layer and the graph convolution layer, we define three architectures of MP-GCN, which are illustrated in Figure 1. In practice, the same as GCN, MP-GCN cannot use too many convolution layers in series to obtain a wider receptive field, when the layer (order) number $l > 2$, its effect is not significantly improved. Therefore, MP-GCN only extracts features

of 1, 2 order neighborhoods. Because our pooling method only focuses on structural information of nodes, so we call it **S-pool**.

In Figure 1, three architectures all use two graph convolution layers (GCNConv by default) to extract features. From up to down, the first *GCNConv* is for aggregating the information of (1-order) nodes immediate adjacent to the central node. The second *GCNConv* is for aggregating the information of the 2-order adjacent nodes. Besides, GCNConv can be replaced by other graph convolution layers, such as ChebConv [5] and GATConv [7]. *Multi-head S-pool* is our proposed pooling layer, which is used to evaluate and select important nodes. Because the information of unselected nodes may be lost during pooling, the residual connection (*Add*) is used to recover their information. *Weight* is used to weight the nodes according to their attention scores calculated by multi-head S-pool. In my model, both *Add* and *Weight* are part of the *Multi-head S-pool*.

1) SELF-ATTENTION-BASED S-POOL

The purpose of the S-pool is to select important nodes correctly and reduce the influence of non-important nodes. It introduces the self-attention mechanism [20] to score the nodes.

Because our short text classification work only uses node structural information, the attention method for calculating element similarity [21] is not suitable. References [22]–[24] use a learnable projection vector p to calculate node scores, which can be expressed as follows:

$$y_i = x_i p / \|p\|, \tag{3}$$

where x_i is the feature vector of node i. y_i is the node score vector and represents the amount of information of node i that can be retained when projected onto the direction of p. Equation (3) is a self-attention method to evaluate the importance of each node.

In our study, the importance can be calculated through 1-dimensional projection:

$$S_A^{(l+1)} = TopK\left(softmax\left(\sigma\left(\tilde{A}H^{(l)}W_A^{(l)}\right)\right)\right), \tag{4}$$

where $H^{(l)} \in \mathbb{R}^{V\times\bar{V}}$ is the hidden state of l layer, and $\bar{V}$ is the node feature dimension. When $l = 0$, $H^{(l)} = H^{(0)} = X$ (identity matrix). $S_A^{(l+1)} \in \mathbb{R}^{V\times 1}$ is the score vector of nodes at $l+1$ layer. W_A is a trainable parameter with the size of $\bar{V}\times 1$, and is generated by the uniform distribution. *TopK* is used to obtain the weights of the top K (selected) nodes with the highest weights and set the weights of other (unselected) nodes to zero. The activation function σ selects *Tanh* for nonlinear stretching. Equation (4) is to score all nodes and realizes the node selection operation of S-pool.

Because the representation of the unlabeled target node can be determined by the selected adjacent node to some extent, so the parameter setting of *TopK* has a certain impact on the performance of the model. To adapt to the change in the number of nodes, we set q as the ratio of the selected node to all nodes, i.e. $K = qV$. Since the number of word nodes is usually greater than the number of document nodes, it can also refer to the vocabulary, i.e. $K = qV_1$, and V_1 is vocabulary size. Fortunately, the significant structures in a graph will not change during computation. Therefore, only one q-value is set for each layer. Through experiments, we notice that the selection of q is mainly influenced by the graph structure.

Through S-pool, MP-GCN can aggregate node information with significant structural characteristics by highlighting important nodes. Besides, it has reasonable complexity and ensures that the whole model is still based on matrix calculation.

2) MULTI-HEAD FOR S-POOL

The S-pool focuses on important but limited nodes. Due to the randomness of initial parameters, some hidden important nodes may be lost. Therefore, the multi-head method [25]–[27] is introduced to form multiple subspaces and enables the model to select nodes from different aspects.

One multi-head method is to convert the size of parameter W_A in (4) to $\bar{V}\times N$. Each column of W_A can be used to calculate a group of node scores. N is the head number. For the stability of the model, we use a more simplified method, which is as follows:

$$\begin{aligned} S_A^{(l+1)} &= RS\left(TopK\left(softmax\left(tanh\left(\tilde{A}XW^{(l)}\right)\right)\right)\right) \\ &= SPool\left(\tilde{A}, W^{(l)}\right), \end{aligned} \tag{5}$$

$$\delta\left(W_A^{(l)}\right) \sim RS\left(\delta\left(W^{(l)}\right)\right), \tag{6}$$

where $S_A \in \mathbb{R}^{V\times N}$ is the attention score matrix. RS is the N-dimensional random sampling operation. δ is the nonlinear transformation function. W is the parameter in convolution calculation (GCNConv). Equation (5) realizes the random sampling of node evaluation (node feature projection) results. What we simplify is the trainable parameters, see parameter replacement in (6). The multi-head evaluation method based on (5) is not as diverse as (4), but its performance is more stable and does not introduce new trainable parameters. This multi-head method enables MP-GCN to learn node representation from different subspaces, which improves the objectivity and stability of node selection.

3) MP-GCN-1

MP-GCN-1 and MP-GCN-2 (**MP-GCN-1/2** for short later) are both used to process isomorphic (text) graphs, and their calculation is similar. We take MP-GCN-1 as an instance to illustrate the calculation of MP-GCN:

$$H_F = \tilde{A}XW^{(0)}, \tag{7}$$

$$S_A^{(1)} = SPool\left(\tilde{A}, W^{(0)}\right), \tag{8}$$

$$S_{Mean}^{(1)} = mean\left(S_A^{(1)}\right), S_{Max}^{(1)} = max\left(S_A^{(1)}\right), \tag{9}$$

$$Z^{(1)} = ReLU\left(H_F \odot \left(1 + S_{Mean}^{(1)} + S_{Max}^{(1)}\right)\right), \tag{10}$$

$$Z^{(2)} = softmax\left(\tilde{A}Z^{(1)}W^{(1)}\right), \tag{11}$$

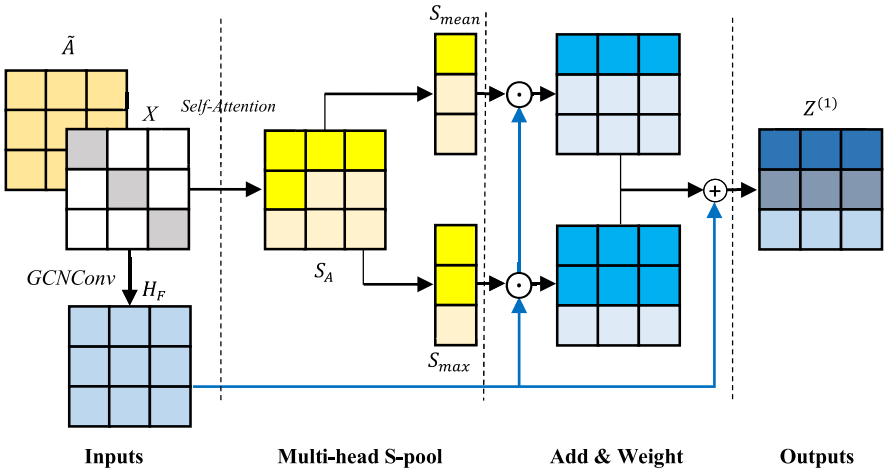


FIGURE 2. An illustration of our pooling layer (multi-head S-pool) with head number N = 3. ⊕ is residual operation. ⊙ is element-wise product. In the figure above, we assume that a graph has three nodes, and each node has three features ($\tilde{A}$ and X). Then, we obtain the attention score matrix (S_A). The mean and maximum of 3 groups of attention scores (S_{max} and S_{mean}) are used to highlight the important nodes and realize node selection. Finally, the enhanced nodes ($Z^{(1)}$) can bring more accurate classification.

where some variables have been explained earlier. The input node feature $X \in \mathbb{R}^{V\times C}$ is initialized to the identity matrix ($C = V$). $\odot$ is element-wise product. *ReLU* is the activation function. $W^{(0)} \in \mathbb{R}^{C\times F}$, $W^{(1)} \in \mathbb{R}^{F\times E}$ are trainable parameters, and F and E are the output dimensions. Equations (7) and (11) are the convolution calculation of the first layer and the second layer, respectively, and $Z^{(1)}$ and $Z^{(2)}$ are their outputs. Equations (8) (9) (10) are for the calculation of the multi-head S-pool. Equation (8) obtains the attention score matrix. In (9), $S_{Mean} \in \mathbb{R}^{V\times 1}$ and $S_{Max} \in \mathbb{R}^{V\times 1}$ are mean and maximum of N groups of attention scores. Equation (10) performs the weighting operation and the residual calculation. An example of multi-head S-pool implementation is shown in Figure 2.

Multi-head S-pool strengthens the representation learning of important nodes. The S-pool highlights the important nodes of the whole graph and weakens some secondary nodes. Introducing the multi-head method can make our network pay attention to the nodes from many aspects.

MP-GCN uses the classical cross-entropy to define the loss function. It does not use the L2 regularization term to constrain the model parameters.

4) GRAPH BUILDING

The graph building affects the performance of the model. Graphs with accurate structure and rich information can bring great help to our task. Common tricks for graph building are as follows:

(1) Adding more relationships (edges) to nodes: Finding more semantic relationships can solve the semantic sparsity of short texts, such as word co-occurrence [2]. In addition, external dependencies can provide more relationships for documents, such as shared topics or entities [3].

(2) Adding multiple types of information to nodes: Building the heterogeneous graph can provide various types of information for nodes, so as to enhance the representation of nodes [29]–[32].

(3) Adding context information to nodes: The graph structure is difficult to represent the semantic and syntactic information in a long continuous word sequence. Combining dependency-based models to obtain context information is a better way to solve this problem.

(4) Combining pre-training model: Bert [15], Glove [33], etc. can provide pre-training word embeddings for node initialization, which can improve the performance of the model in many NLP tasks.

In our study, we propose MP-GCN-1* and apply it to process heterogeneous (text) graphs. It can realize text classification by learning multiple representations with different types of nodes without increasing model parameters.

First, inspired by [2], we build a heterogeneous graph for the corpus and turn the text classification problem into the node classification problem. The nodes of this graph are composed of documents and words. Because the documents can be represented by the sum of word embedding vectors, the method of processing the isomorphic graph can be directly used in this graph.

This graph focuses on global word co-occurrence information in a corpus [2]. Its corresponding adjacency matrix is defined as:

$$A^C = \begin{cases} PMI\ (i,j) & i,j\ are\ words \\ TF-IDF & i\ is\ word, j\ is\ document \\ 1 & i=j \\ 0 & otherwise, \end{cases} \tag{12}$$

where A^C denotes the connection relationship between nodes i and j. The edge weights between word nodes are calculated by PMI [2], which explicitly models word co-occurrence. The edge weights between document

TABLE 1. Summary statistics of datasets [2] and model parameter setting.

Name	#Docs	#Nodes	#Class	Average Length	Edge	q
MR	10,662	29,426	2	20.39	1,701,424	0.05*V_1
R8	7,674	15,362	8	65.72	3,165,430	0.01*V_1
R52	9,100	17,992	52	69.82	3,981,246	0.01*V_1
Ohsumed	7,400	21,557	23	135.82	7,456,448	0.01*V_1
20NG	18,846	61,603	20	221.26	24,689,966	0.01*V_1

(sentence) nodes and word nodes are calculated by TF-IDF, which models the importance of words in documents. The diagonal element of A^C is set to 1. In the following experiments, A^C is the input of MP-GCN-1/2. A^C has two types of edges. Although it can be treated as an isomorphic graph [2], it is essentially a heterogeneous graph.

However, this operation of treating a heterogeneous graph as an isomorphic graph may not be optimal. Therefore, different types of edges should be treated separately. Specifically, we set the adjacency matrix containing only the edges between word nodes as A^P and the adjacency matrix containing only the edges between word nodes and document nodes as A^T. A^C and A^P can be obtained by masking operation. A^C, A^P, and A^T are the inputs of MP-GCN-1*.

5) MP-GCN-1*

Different from the previous method (e.g. RGCN [32]) of dealing with heterogeneous graphs, MP-GCN-1* does not introduce new trainable parameters. Its implementation differs from MP-GCN-1 in the following aspects:

$$H_F = A^C X W^{(0)}, \tag{13}$$

$$S_P^{(1)} = SPool\left(A^P, W^{(0)}\right), \tag{14}$$

$$S_T^{(1)} = SPool\left(A^T, W^{(0)}\right), \tag{15}$$

$$S_{Mean}^{(1)} = mean\left(S_P^{(1)}\right) + mean\left(S_T^{(1)}\right), \tag{16}$$

$$S_{Max}^{(1)} = max\left(S_P^{(1)}\right) + max\left(S_T^{(1)}\right). \tag{17}$$

Compared to MP-GCN-1/2, MP-GCN-1* still does not increase trainable parameters. It evaluates the importance of nodes according to different types of edges, which makes the model more stable and accurate.

IV. EXPERIMENT

A. DATASET

We ran our experiments on five widely used benchmark corpora[1] including 20-Newsgroups[2] (20NG), Ohsumed,[3] R52 and R8 of Reuters dataset,[4] and Movie Review (MR)[5] [2]. These datasets were processed by cleaning data, segmentation, removing stop words, and removing words whose frequency is less than 5. Some important descriptions of datasets are listed in Table 1.

[1] https://github.com/yao8839836/text_gcn/tree/master/data
[2] http://qwone.com/~jason/20Newsgroups/
[3] http://disi.unitn.it/moschitti/corpora.htm
[4] http://www.daviddlewis.com/resources/testcollections/reuters21578/
[5] https://github.com/mnqu/PTE/tree/master/data/mr

B. BASELINE

Baselines can be categorized into four categories, including:

(1) **Traditional model**: TF-IDF+LR [2].

(2) **Word embedding models**: PV-DBOW [34], PV-DM [34], FastText [35], SWEM [36], and LEAM [37].

(3) **Dependency-based deep learning models**: CNN [13] and LSTM [14].

(4) **Graph-based deep learning models**: Graph-CNN-C [5], Graph-CNN-S [5], Graph-CNN-F [2], TextGCN [2].

To conduct a fair comparison study, all models did not use pre-trained word embedding. They utilized the same settings in [2], [9], [11].

C. SETTINGS

The main parameter settings of MP-GCN included: the head number $N = 12$, the learning rate $e = 0.005$, and the ratio of Top-K $q = kV_1$ (see Table 1).

D. RESULTS ANALYSIS

1) PERFORMANCE

Table 2 shows that, the traditional model achieves better performance than the dependency-based models. The word embedding models are superior to the two models because of the joint representation of word context. Benefit from the rich representation of the graph, graph-based models surpass the above models. Among them, the performance of Text-GCN is outstanding. Without pre-training word embedding, MP-GCN significantly outperforms all baselines and achieves state-of-the-art results on these benchmark datasets. In the three architectures of MP-GCN, the performance of the MP-GCN-1/2 is similar. MP-GCN-1* outperforms MP-GCN-1/2 on four datasets.

The reasons why MP-GCN works well include that: 1) it is mainly due to the advantages of graph structure, the rich links effectively associate the nodes (including sparse ones), so that these nodes can be effectively represented. 2) MP-GCN can enhance the representation learning of important nodes. These selected and enhanced nodes contain

TABLE 2. Test accuracy comparison with baselines on benchmark datasets. All models ran 10 times and were report by mean ± standard deviation.

Model	MR	R8	R52	Ohsumed	20NG
TF-IDF+LR	0.7459±0.0000	0.9374±0.0000	0.8695±0.0000	0.5466±0.0000	0.8319±0.0000
CNN	0.7498±0.0070	0.9402±0.0057	0.8537±0.0047	0.4387±0.0100	0.7693±0.0061
LSTM	0.7506±0.0044	0.9368±0.0082	0.8554±0.0113	0.4113±0.0117	0.6571±0.0152
PV-DBOW	0.6109±0.0010	0.8587±0.0010	0.7829±0.0011	0.4665±0.0019	0.7436±0.0018
PV-DM	0.5947±0.0038	0.5207±0.0004	0.4492±0.0005	0.2950±0.0007	0.5114±0.0022
FastText	0.7217±0.0130	0.8604±0.0024	0.7155±0.0042	0.1459±0.0000	0.1138±0.0118
FastText (bigrams)	0.6761±0.0279	0.8295±0.0003	0.6819±0.0004	0.1459±0.0000	0.0734±0.0068
SWEM	0.7665±0.0063	0.9532±0.0026	0.9294±0.0024	0.6312±0.0055	0.8516±0.0029
LEAM	0.7695±0.0045	0.9331±0.0024	0.9184±0.0023	0.5858±0.0079	0.8191±0.0024
Graph-CNN-C	**0.7722±0.0027**	0.9699±0.0012	0.9275±0.0022	0.6386±0.0053	0.8142±0.0032
Graph-CNN-S	0.7699±0.0014	0.9680±0.0020	0.9274±0.0024	0.6282±0.0037	--
Graph-CNN-F	0.7674±0.0021	0.9689±0.0006	0.9320±0.0004	0.6304±0.0077	--
Text-GCN[1]	0.7674±0.0020	**0.9707±0.0010**	**0.9356±0.0018**	**0.6836±0.0056**	**0.8634±0.0009**
MP-GCN-1	0.7763±0.0013	0.9771±0.0009	0.9370±0.0027	0.7007±0.0014	0.8692±0.0017
MP-GCN-1*	**0.7792±0.0010**	**0.9785±0.0016**	**0.9454±0.0008**	**0.7027±0.0012**	0.8684±0.0013
MP-GCN-2	0.7766±0.0016	0.9778±0.0017	0.9430±0.0012	0.6975±0.0032	**0.8702±0.0008**

more distinctive features, which can make the classification more accurate.

Besides, the processing effect of our model on longer text is limited (see 20ng in Table 2), and the benefit of increased computational consumption is low. Because it is difficult to select a small number of important nodes as the semantic representation of the whole long text document.

2) ABLATION ANALYSIS

In Table 3, it can be seen from the experiment that the performance of our models will be greatly reduced without using any of the methods of multi-head or TopK. To achieve better results, our models must combine the two methods for pooling.

TABLE 3. Ablation experiment of MP-GCN on MR. This experiment analyzes the performance comparison of MP-GCN when using only the single head (-MH) and not using the TopK method to select nodes (-TOPK).

Model	-MH	-TOPK	Baseline
MP-GCN-1	0.7660±0.0014	0.7616±0.0046	**0.7763±0.0007**
MP-GCN-1*	0.7640±0.0025	0.7631±0.0035	**0.7792±0.0010**
MP-GCN-2	0.7663±0.0020	0.7636±0.0031	**0.7766±0.0011**

In Table 4, the experimental results show that different types of edges bring different improvements to the model. As the reference information for evaluating the importance of nodes, the edge from A^P is more informative than the edge from A^T. The experimental results also confirm the effectiveness of the model for heterogeneous graph processing.

Through the above two ablation experiments, it is found that the worst performance of our models is close to the Text-GCN. This is because the residual connection is added to the model architecture, which makes the model more stable and will not significantly reduce the performance due to the failure of the trick.

TABLE 4. Ablation experiment of MP-GCN-1*. This experiment analyzes the performance comparison of MP-GCN-1* without using edge information between word nodes ($-A^P$) or edge information between word nodes and document nodes ($-A^T$).

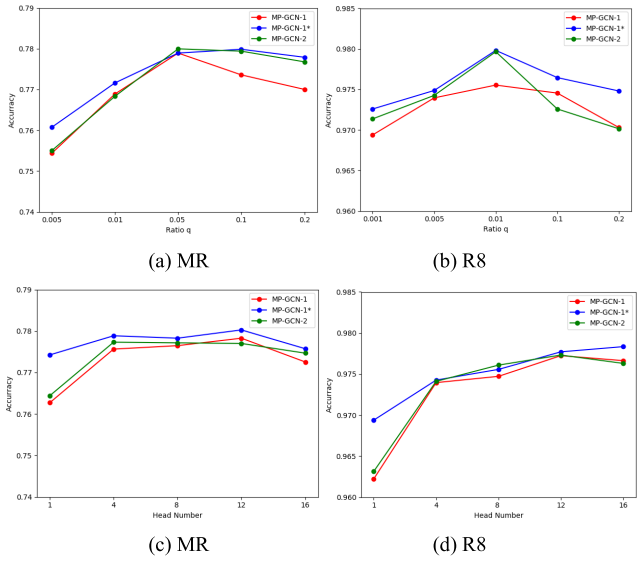
Dateset	$-A^P$	$-A^T$	A^C
MR	0.7681±0.0009	0.7710±0.0013	**0.7792±0.0010**
R8	0.9725±0.0027	0.9764±0.0008	**0.9785±0.0016**
R52	0.9390±0.0010	0.9412±0.0016	**0.9454±0.0008**

3) PARAMETER SENSITIVITY

Figures 3a and 3b show that MP-GCN can get better performance by adjusting the q-value. The selection of q is mainly related to the graph structure, and its setting in the same layer of MP-GCN is constant. Figures 3c and 3d show that excess heads may reduce the efficiency and the effect. But with insufficient heads, MP-GCN will only focus on fewer subspaces, which reduces the accuracy and objectivity of node selection. Through repeated experiments, it is proved that MP-GCN has a more stable performance when the head number $N = 12$.

4) EFFECTS OF THE SIZE OF LABELED DATA

Since our model is a semi-supervised model, we also tested the performance of the model under different proportions of training data. Figures 4 reports test accuracies with 2.5%,



(a) MR (b) R8

(c) MR (d) R8

FIGURE 3. Parameter sensitivity. Figures (a) (b) describe the influence of the ratio q of TopK on the accuracy. Figures (c) (d) describe the influence of the head number on the accuracy.

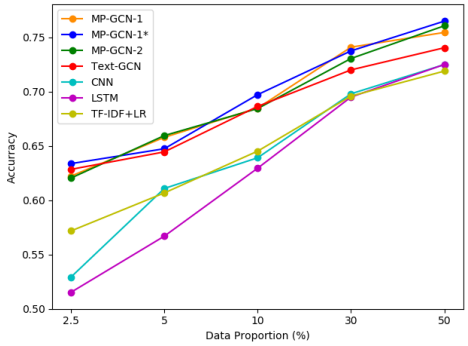


FIGURE 4. Test accuracy by varying training data proportions.

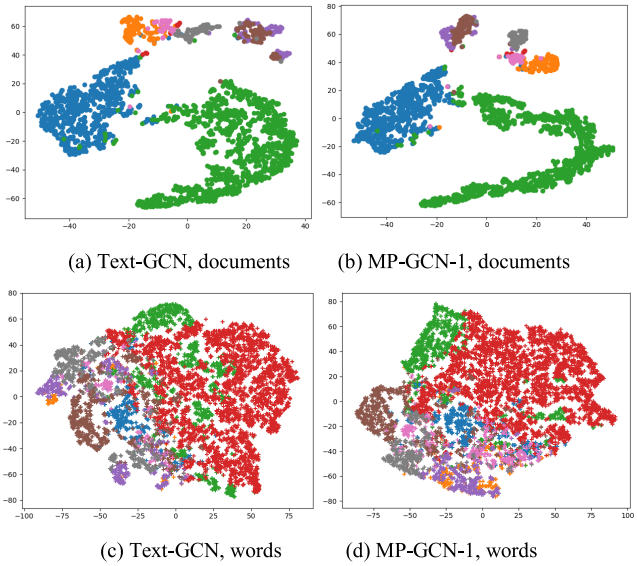
5%, 10%, 30% and 50% of MR training set. Compared with some baseline models,[6] our three models and Text-GCN can achieve higher test accuracy with limited labeled documents. It is because these GCN-based models can propagate document label information to the entire graph well [2].

5) TIME CONSUMPTION

With the introduction of the multi-head S-pool, the computational complexity of the MP-GCN will inevitably increase compared with GCN. This is because it does not do down-sampling like traditional pooling methods, but enhances the representation learning of selected nodes without deleting any nodes.

Taking the main equations in MP-GCN-1 as an instance, the computational complexity formulas of (7), (8), and (12) are $O(|\varepsilon| CF)$, $O(|\varepsilon| CF)$, and $O(|\varepsilon| FE)$, respectively. ε is the set of edges and $|\varepsilon|$ is the number of edges. After integration, the maximum computational complexity of

[6] https://github.com/shanzhonglujie/MP-GCN/tree/main/test



(a) Text-GCN, documents (b) MP-GCN-1, documents

(c) Text-GCN, words (d) MP-GCN-1, words

FIGURE 5. t-SNE visualization of the second layer document and word embeddings learned from R8. In the figures above, different colors mean different document classes. In Figures (c) (d), we set the dimension with the largest value as a word's label [2].

MP-GCN-1 is close to that of two-layer GCN [1], i.e., $O(|\varepsilon| CFE)$ or $O(2|\varepsilon| CF + |\varepsilon| FE)$ ($C = V$, $V \ll |\varepsilon|$). Compared with two-layer GCN, MP-GCN-1 mainly increases the acceptable computational cost of (8) (9) (10). Through the above analysis, it can be seen that the time consumption of MP-GCN is still mainly related to the number of edges. Besides, we build a graph based on the whole corpus, and the memory consumption is $O(|\varepsilon|)$.

6) VISUALIZATION

The MP-GCN can output better graph embeddings of words and documents. We visualized them and compared the performance of node representation learning between Text-GCN [2] and MP-GCN. Figures 5a and 5b show that the document node embedding of MP-GCN-1 has a higher vector aggregation degree, i.e., MP-GCN-1 can learn more discriminative document embeddings. Figures 5c and 5d show that the node embedding of MP-GCN-1 contains more words with the same labels and close position (similar semantics), which means most of them are more closely related to certain document classes [2]. Experimental results illustrate that the output embedding of MP-GCN-1 is more discriminative than that of Text-GCN. Besides, other MP-GCN models can achieve the same effect.

7) COMPARED WITH THE PRE-TRAINING MODELS

Text classification is a regular task of NLP. In this task, Bert has achieved a dominant position. In our study, we introduced the Bert module into MP-GCN and compared it with the Bert-based models. The experimental results are shown in Table 5. BertGCN[7] is the text classification model combining Bert

[7] https://github.com/MorningForest/BertGCN

and GCN, and it achieves the SOTA performance. Details for the baseline models and dataset can be found in [38]. In the experiment, the head number of our model was set to 1. The source code of our Bert-based MP-GCN is also available.[8]

TABLE 5. Test accuracy comparison with the Bert-based models. We run the models 10 times and report the mean test accuracy.

Model	MR	R8	R52
Bert	85.7	97.8	96.4
BertGCN	86.0	98.1	96.6
Bert+MP-GCN	**86.3**	**98.3**	**96.8**

Combined with the two baselines, the Bert-based MP-GCN performs better. The proposed pooling method plays a guiding role in training. Compared with BertGCN, our model can achieve good performance with fewer iterations, and it is easier to converge, see Figure 6. In this experiment, our model can get better results within 10 iterations.

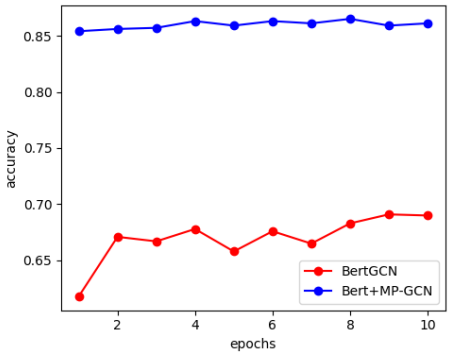


FIGURE 6. Accuracy comparison of training data on dataset MR. The main parameter settings of the two models are the same.

Combined with the pre-training model (such as Bert), the performance of our model can be improved to a certain extent. However, because the grammar and semantics of the text in the pre-training must be consistent with the downstream task, the pre-training models cannot be applied to all scenarios, especially specific scenes. In contrast, MP-GCN can build a graph to solve the text classification problem without pre-training word embedding, i.e., it builds a new language model. Therefore, our model can be applied in most specific scenarios.

8) DISCUSSION

MP-GCN is an innovation of network structure based on GCN and has a stronger ability to cover all data. When applied to short text classification, it can provide a certain degree of attention for long-tailed (sparse) words.

Why does MP-GCN mainly focus on the first-order nodes? Through experiments, it is found that the structure information of the first-order nodes extracted by GCN is more important than that of the second-order nodes. Similar experiments can be found in [39].

Why does MP-GCN not pool the input of the second graph convolution layer? If pooling is used, the classification effect of the model will not necessarily increase, but the computational consumption will increase.

[8] https://github.com/shanzhonglujie/MP-GCN/tree/main/bert-based

V. CONCLUSION

In this study, we propose MP-GCN for short text classification. This network introduces multi-head pooling to enhance the representation learning of important nodes. We introduce three architectures of MP-GCN, which focus on node representation learning of 1-order, 1&2-order of isomorphic graph, and 1-order of heterogeneous graph, respectively. Experimental results demonstrate that, without using pre-training embedding, MP-GCN can outperform state-of-the-art models across five benchmark datasets.

REFERENCES

[1] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*.
[2] L. Yao, C. Mao, and Y. Luo, "Graph convolutional networks for text classification," in *Proc. Conf. Assoc. Adv. Artif. Intell. (AAAI)*, Sep. 2019, pp. 7370–7377.
[3] H. Linmei, T. Yang, C. Shi, H. Ji, and X. Li, "Heterogeneous graph attention networks for semi-supervised short text classification," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*, 2019, pp. 4821–4830.
[4] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," 2013, *arXiv:1312.6203*.
[5] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. Conf. Neural Inf. Process. Syst. (NeurIPS)*, Jun. 2016, pp. 3844–3852.
[6] W. Hamilton, Z. T. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Conf. Neural Inf. Process. Syst. (NeurIPS)*, Dec. 2017, pp. 1024–1034.
[7] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," 2017, *arXiv:1710.10903*.
[8] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," 2018, *arXiv:1812.08434*.
[9] L. Huang, D. Ma, S. Li, X. Zhang, and H. Wang, "Text level graph neural network for text classification," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*, 2019, pp. 3444–3450.
[10] C. Zhang, Q. Li, and D. Song, "Aspect-based sentiment classification with aspect-specific graph convolutional networks," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*, Nov. 2019, pp. 4568–4578.
[11] X. Liu, X. You, X. Zhang, J. Wu, and P. Lv, "Tensor graph convolutional networks for text classification," in *Proc. Conf. Artif. Intell. (AAAI)*, Apr. 2020, pp. 8409–8416.
[12] H. Peng, J. Li, Y. He, Y. Liu, and M. Bao, "Large-scale hierarchical text classification with recursively regularized deep graph-CNN," in *Proc. Conf. World Wide Web (WWW)*, Apr. 2018, pp. 1063–1072.
[13] Y. Kim, "Convolutional neural networks for sentence classification," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, Oct. 2014, pp. 1746–1751.
[14] T. Liu, X. Zhang, W. Zhou, and W. Jia, "Neural relation extraction via inner-sentence noise reduction and transfer learning," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Oct. 2018, pp. 2195–2204.
[15] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.
[16] H. Gao and S. Ji, "Graph U-Nets," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Jul. 2019, pp. 741–749.

[17] R. Ying, J. You, C. Morris, X. Ren, W. L. Hamilton, and J. Leskovec, "Hierarchical graph representation learning with differentiable pooling," 2018, *arXiv:1806.08804*.
[18] Y. Ma, S. Wang, C. Aggarwal, and J. Tang, "Graph convolutional networks with eigenpooling," in *Proc. Knowl. Discovery Data Mining (KDD)*, Jul. 2019, pp. 723–731.
[19] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Jan. 2021.
[20] C. Zheng, X. Fan, C. Wang, and J. Qi, "GMAN: A graph multi-attention network for traffic prediction," in *Proc. Conf. Assoc. Adv. Artif. Intell. (AAAI)*, Apr. 2020, pp. 1234–1241.
[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017, *arXiv:1706.03762*.
[22] L. Zhang, X. Wang, H. Li, G. Zhu, P. Shen, P. Li, X. Lu, S. A. A. Shah, and M. Bennamoun, "Structure-feature based graph self-adaptive pooling," in *Proc. Conf. World Wide Web (WWW)*, Apr. 2020, pp. 3098–3104.
[23] J. Lee, I. Lee, and J. Kang, "Self-attention graph pooling," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Apr. 2019, pp. 3734–3743.
[24] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," 2014, *arXiv:1404.2188*.
[25] E. Voita, D. Talbot, F. Moiseev, R. Sennrich, and I. Titov, "Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics (ACL)*, Aug. 2019, pp. 5797–5808.
[26] J. Vig, "A multiscale visualization of attention in the transformer model," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics, Syst. Demonstrations (ACL)*, Aug. 2019, pp. 37–42.
[27] A. Raganato and J. Tiedemann, "An analysis of encoder representations in transformer-based machine translation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, Novl. 2018, pp. 287–297.
[28] Y. Ding, Y. Liu, H. Luan, and M. Sun, "Visualizing and understanding neural machine translation," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics (ACL)*, Jul. 2017, pp. 1150–1159.
[29] L. Zheng, Z. Li, J. Li, Z. Li, and J. Gao, "AddGraph: Anomaly detection in dynamic graph using attention-based temporal GCN," in *Proc. Joint Conf. Artif. Intell. (IJCAI)*, Aug. 2019, pp. 4419–4425.
[30] H. Zhang and J. Zhang, "Text graph transformer for document classification," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, Nov. 2020, pp. 8322–8327.
[31] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, and P. S. Yu, "Heterogeneous graph attention network," in *Proc. World Wide Web Conf. (WWW)*, May 2019, pp. 2022–2032.
[32] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. van den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," 2017, *arXiv:1703.06103*.
[33] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, Oct. 2014, pp. 1532–1543.
[34] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *Proc. Int. Conf. Mach. Learn. (ICML), Jun.*, vol. 2014, pp. 1188–1196.
[35] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of tricks for efficient text classification," in *Proc. 15th. Int. Conf. Eur. Chapter Assoc. Comput. Linguistics (EACL)*, Apr. 2017, pp. 427–431.
[36] D. Shen, G. Wang, W. Wang, M. Renqiang, and L. Carin, "Baseline needs more love: On simple word-embedding-based models and associated pooling mechanisms," in *Proc. Annu. Meeting Assoc. Comput. Linguistics (ACL)*, Jul. 2018, pp. 440–450.
[37] G. Wang, C. Li, W. Wang, Y. Zhang, and D. Shen, "Joint embedding of words and labels for text classification," in *Proc. Annu. Meeting Assoc. Comput. Linguistics (ACL)*, Jul. 2018, pp. 2321–2331.
[38] Y. Lin, Y. Meng, X. Sun, Q. Han, K. Kuang, J. Li, and F. Wu, "BertGCN: Transductive text classification by combining GCN and BERT," 2021, *arXiv: 2105.05727*.
[39] Y. Zhang, P. Qi, and C. D. Manning, "Graph convolution over pruned dependency trees improves relation extraction," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, Oct. 2018, pp. 2205–2215.

HONGYU ZHAO received the Ph.D. degree in computer application technology from the Beijing University of Technology, Beijing, China, in 2015. He is currently a Joint Postdoctor of the Southwest University and Mashang Consumer Finance Company Ltd. His research interests include natural language processing, computer vision, and image processing.

JIAZHI XIE received the Ph.D. degree in rural finance and fiscal from Southwest Agricultural University, Chongqing, China, in 2001. He is currently a Professor with Southwest University, Sichuan, China. His research interests include financial economics, commercial insurance theory and practice, and risk management theory research.

HONGBIN WANG received the Ph.D. degree in computer application technology from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2003. He is currently the Artificial Intelligence Institute Director of Mashang Consumer Finance Company Ltd., Chongqing, China. His research interests include natural language processing, computer vision, and speech recognition.