Received 1 November 2022, accepted 15 November 2022, date of publication 28 November 2022, date of current version 5 December 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3225445

RESEARCH ARTICLE

Multilabel Text Classification Using Multilayer DGAT

HUI CHEN[1], **(Member, IEEE), JIAN HUANG**[2], **NANA TAO**[2], **JIJIE HUANG**[1], **AND JING WANG**[2], **(Member, IEEE)**
[1]School of Electronic and Information Engineering, Foshan University, Foshan 528225, China
[2]School of Software and IoT Engineering, Jiangxi University of Finance and Economics, Nanchang 330012, China

Corresponding author: Jian Huang (hjian@jxufe.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61962024 and Grant 61866014, and in part by the Scientific Research Foundation of Education Department of Jiangxi Province of China under Grant GJJ190258 and Grant GJJ190254.

ABSTRACT Text classification is one of the most fundamental tasks in text data analysis. Generally, textual data are extensive in scale and complex in the inherent relationship, leading to low accuracy in traditional classification models. This paper proposes a multi-label classification model based on multi-layer neural network architecture. We first construct a dual-attention mechanism graph neural network (named DGAT for short) to fuse the typological and informational features of the target node and the connected nodes. Secondly, we also build a multi-layer network architecture with multiple DGATs to expand the range of neighborhood nodes participating in the feature fusion to meet the needs of classifying different datasets. To ensure the learning ability of the model, we also use the residual network to solve the problem of error rise and gradient descent caused by multi-layer network architecture. Finally, we conducted a large number of experiments on five benchmark datasets. The results show that the accuracy of the proposed model is significantly better than that of the traditional models, and that there is a noticeable improvement when compared with other deep learning methods.

INDEX TERMS Multilabel text classification, feature fusion, graph attention network, residual network.

I. INTRODUCTION

As internet technology enters the Web 2.0 era, the data on the internet are exploding. While people enjoy the convenience brought by massive data, it is difficult to retrieve and acquire data because of the complexity and diversity of data. Text classification is an important method that can help users organize data efficiently and improve the speed and quality of data retrieval. Text classification is not only an essential branch of data mining, but also a classic problem in natural language processing [1]. It has a wide range of applications, such as language translation, text summarization, news recommendation, and spam filtering. Text classification uses natural language processing, data mining, and machine learning to effectively classify different text types and discover rules [2]. However, there are rarely tags in data in practice, and manually generating tagged data will consume time and cause low accuracy. Therefore, it is particularly urgent to conduct semi-supervised text classification research under the condition of a small amount of labeled data.

Text classification extracts semantic features from the original text corpus and predicts the topic categories of text data based on these features. In the past few decades, various models for text classification have emerged. Early text classification methods mainly rely on the classification rules manually discovered and formulated by domain experts. With the emergence of statistical learning and machine learning, text classification based on machine learning has made a breakthrough. Common machine learning-based classifiers include support vector machine [3], naive Bayes [4], K-nearest neighbor [5], decision tree [6], and logistic regression [7]. Compared with earlier rule-based methods, text classification methods based on statistical models have apparent advantages in terms of accuracy and stability. However, these methods still require

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

laborious and time-consuming feature engineering. In addition, they typically do not consider natural sequential structures or contextual information. This makes it difficult for models to learn semantic information.

After the 2010s, text classification methods gradually changed from statistical to deep learning methods. The deep learning model uses an artificial neural network that simulates the way human brains to classify data. Compared with statistical model-based methods, deep learning methods avoid artificial design rules and features and can automatically mine a large number of rich semantic representations from the text.

Feedforward neural networks and Recursive Neural Networks (RNNs) were the first two deep learning methods for text classification tasks. Many researchers have improved the performance of text categorization for different tasks by improving Convolutional Neural Networks (CNNs), RNNs, attention mechanisms, model fusion and multitask methods. However, the text classification method based on CNN and RNN is implemented on the premise of independent texts. There are correlation relationships among the texts, such as citation networks, social relationship networks, and biomolecular structures. This correlation has an important role in the discrimination of text categories. Therefore, researchers pay more attention to designing more efficient text classification method by combining graph neural networks with convolutional neural networks and attention mechanisms. Kipf et al. proposed a semi-supervised text classification method [8] based on a graph convolutional neural network (GCN) for classifying the citation datasets. Yao et al. proposed the TextGCN [9] model for classifying text datasets without clear correlation. Wu et al. designed a multilayer convolutional neural network model named SGC [10] for text classification.

Compared with neural network models such as CNN and RNN, the GCN model significantly improves classification accuracy. However, it also has obvious shortcomings. First, the lack of flexibility of the GCN model leads to poor scalability. Second, GCN model training is full-batch, which is difficult to extend to large-scale networks and has slow convergence. In addition, in the GCN model, the weight of adjacent node features depend on the graph's structure and has nothing to do with the features of the node itself, which is contrary to the goal of text classification. In contrast to this, graph attentional Neural Network (GAT) [11] takes the advantages of GCN into account. Meanwhile, the weight of adjacent node features in GAT entirely depends on the node features, independent of the graph's structure. These advantages make GAT more suitable for solving text classification problems.

To overcome the problems of large-scale text data classification, this paper designs a semi-supervised multilabel text classification model based on a multi-layer DGAT. First, this model allows text features to be propagated along the correlation relationship and fused with the associated text features to improve the text feature expression ability. Second, we design a dual-level attentional model to distinguish the influence of associated nodes on target nodes. Furthermore, we apply network stacking and residual connections to improve the range of information transmission and ensures the learning ability of the model. Given a small amount of labeled text and text association graphs, the classification model can classify unlabeled text. The main contributions of this paper are as follows:

- In this paper, we propose a feature fusion framework based on DGAT. In this framework, we can combine multi-layer DGAT and residual network to build a feature fusion framework for nodes connected across multiple hops. This framework can improve the expression ability of node features and reduce the risk of model overfitting.
- Using the idea of information transmission, we aggregate the features of nodes in the local neighborhood to the target node and distinguish the weight from the two aspects of the node's type and its eigenvalue. In addition, we also control the range of neighborhood nodes by changing the number of DGAT layers. It can effectively improve the model's accuracy and the ability to deal with large-scale text data.
- The proposed method is implemented and compared with many existing works on five benchmark datasets. Experimental results show it has clear advantages in terms of model correctness.

II. PROBLEM DEFINITION AND RELATED WORK

This section first formulates the problem and then summarizes the relevant work on text classification based on graph attention networks.

A. PROBLEM DEFINITION

Let $D = \{d_1, d_2, \cdots, d_n\}$ represent the citation dataset composed of n documents, and C represent the class set of D. For any two documents, if there is a reference relationship between them, we. consider there to be an edge between them. Thus, a text infographic with documents as the vertex set and a reference relationship as the edge set can be obtained.

For any document $d_i \in D$, its document class is denoted by c_{di}. Then, $c_{di} \in C$ if d_i is labeled; otherwise, $c_{di} = null$. We aim to build a neural network model with a dual attention mechanism for classifying unlabeled documents by learning document features and the association between documents.

B. DEEP LEARNING FOR TEXT CLASSIFICATION

In recent years, deep learning has made a breakthrough in text, images, and other fields. Compared with text classification models of traditional machine learning, deep learning methods can obtain better semantic representation and reduce incompleteness and complexity in the feature extraction process. The research on deep learning text classification can be divided into two categories. One is the word embedding model studied by Mikolov et al., which aggregates and embeds unsupervised words into documents and then embeds these documents into the classifier. The other is to use neural networks, such as CNNs [12]. CNN is a deep neural network with a convolutional structure. Convolutional design can reduce the memory requirements of deep text classification methods and can be roughly divided into four categories: reinforcement learning, ensemble learning, transfer learning, and deep learning. This section summarizes the characteristics of the method proposed in this article based on the analysis of related work.

In terms of applying convolutional neural networks to text classification tasks, Kim et al. proposed a model using CNN. This model trains a neural network model using labeled data to predict data categories. It includes a word vector layer, convolutional layer, top pooling layer, fully connected layer, and a softmax layer. There are many hyperparameters to choose from in the convolutional neural network. Gori et al. [13] conducted various comparative experiments under different hyperparameter settings for the model in the literature and gave suggestions for parameter tuning and location experience. Many researchers have studied and improved the model proposed by Kim et al. regarding structure, depth, and training speed. For example, Wang et al. [14] performed semantic feature extraction in the word vector matrix before inputting the convolutional layer and finally performing classification. Kalchbrenner et al. [15] proposed a network model called a DCNN (dynamic convolutional neural network), which uses a multilayer convolutional neural network and can handle variable-length inputs. However, these methods are all external neural networks and consider using deep neural networks to solve the text classification problem later. For example, the DPCNN (deep pyramid convolutional neural networks) [16] proposed by Tencent AI-lab in ACL2017 extracts the text's long-distance dependence problem by deepening the number of layers of the neural network.

Although these methods are effective, they mainly focus on local word sequences and do not focus on the corpus' global word co-occurrence information.

C. SEMI-SUPERVISED TEXT CLASSIFICATION

Due to the lack of annotated datasets and the inaccuracy of manual annotations, semi-supervised methods have been proposed. They can be categorized into two classes: (1) latent variable models and (2) embedding-based models [17]. The former extends the topic model through user-provided seed information and then infers the documents' labels based on posterior category-topic assignment. The latter uses seed information to export the embedding of the documents and label names for text classification. For example, Yin et al. [18] used a semi-supervised learning method based on SVM to label unlabeled documents iteratively. GCNs have recently achieved good results in semi-supervised classification and have received extensive attention.

D. GRAPH CONVOLUTIONAL NETWORKS

Scarselli et al. [19] were the first to try to extend the neural network to graph structures [13], [20], but it did not initially attract wide attention. CNNs have made extraordinary progress in computer vision and other fields, opening a new era of deep learning [21], [22]. The critical points of CNN are local connection, weight sharing, and the use of a deep network [23]. However, the pixel points in the image or video data processed by it are neatly arranged matrices and Euclidean structures. The corresponding data of a non-Euclidean structure, such as a social network, is the topology diagram in the abstract sense in graph theory, and the CNN finds it challenging to deal with.

In the last few years, there has been much interest in using convolution on graphs. These methods are based on neighborhood aggregation schemes and can be further divided into spectral methods and spatial methods. Spectral methods are based on spectral theory to define parametric filters. Bruna et al. [24] first defined a parametric filter in the Fourier domain. However, its large amount of computation limits its scalability [25]. To improve efficiency, Defferrard et al. proposed the Chebnet algorithm and approximated the K-polynomial filter through the Chebyshev expansion of the Laplacian operator. Kipf and Welling further simplified the Chebnet algorithm by simplifying Chebyshev polynomials.

The spatial method combines the neighborhood information of the vertex domain to generate node embedding. MoNet [26] and SplineCNN [27] integrate local signals by designing an operator. In the literature [8], Thomas N. Kipf and Max Welling proposed a scalable semi-supervised learning method based on a graph data structure GCN multilayer neural network that runs directly on the graph. The model scales linearly on the number of edges in the graph and can learn hidden layer representations, which encode both the local graph structure and node features. Later, Henaff et al. [28] explored graph neural networks for text classification. However, these models treat documents or sentences as nodes or rely on standardized literature citation relationships to build graphs, which have significant limitations. Deferred et al. [29] and Peng et al. [30]. have explored text classification algorithms based on graph convolutional networks. Recently, Yao et al. [9] used GCN for text classification. Their model is named TextGCN. The model embeds the entire corpus into a graph, which uses documents and words as nodes and links between documents and words and between words as edges. Given a predefined weight for each edge, the GCN model can be trained using the labeled data, and the unlabeled can be predicted based on the model. Gao et al. select a fixed number of neighborhood nodes for each feature and allow conventional convolution operations on Euclidean spaces.

III. FEATURE FUSION BASED ON TEXT INFORMATION GRAPH

This section presents text datasets and reference relationships using text infographics and discusses the correlation between texts and text feature fusion methods.

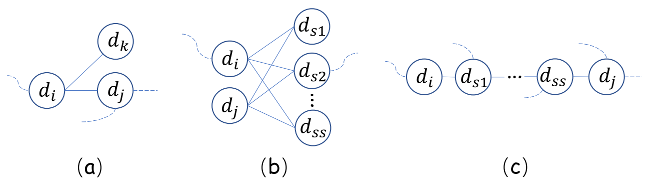

FIGURE 1. **Possible text association relationships based on text references.**

A. TEXT INFORMATION GRAPH

Given a text set $D = \{d_1, d_2 \ldots, d_n\}$ with reference relation, a weighted graph $G = (V, E, W)$ can be used to represent text set D and its relations. The texts in D constitute the vertex set of graph G and there is a one-to-one correspondence relationship between the texts and the vertices. That is, for any text $d_i \in D$ uniquely corresponds to a vertex $v_i \in V$, and vice versa. Similarly, the referential relationships between texts in D form edges in graph G. If text d_i references text d_j, there exists an edge from vertex v_i to vertex v_j in graph G, that is, $e_{ij} = 1$, and w_{ij} represents the weight of edge e_{ij}. Initially, w_{ij} is set to 1. For any vertex v_i in graph G, the set of its adjacent vertices is denoted as $N(v_i)$.

According to the reference relationship between texts, as shown in Figure 1, there may be the following possible relations between any two texts d_i and d_j: (1). As shown in Figure 1(a), there is a direct reference relationship between d_i and d_j; (2). d_i and d_j do not have direct references, but they are related to the same neighbors. $d_{s1}, \cdots, d_{ss}$ has a reference relation, as shown in Figure 1(b); (3). As shown in Figure 1(c), the texts d_i and d_j passing text $d_{s1}, \cdots, D_{ss}$ have an indirect reference relationship; (4) d_i and d_j have the complex relationship of multiple references mentioned above.

B. INFORMATION TRANSMISSION AND FEATURE FUSION

In a text infographic, the reference relation between texts not only represents the correlation relation between texts but also contains the direction of text feature fusion. If there are reference relations between text d_i and its neighbor nodes, then we can use vertex v_i to represent the feature fusion of the features of text d_i and its neighbor nodes. Let vector $h_i \in \mathcal{R}^m$ denote the feature of text d_i, and let $N(v_i)$ represent the set of vertices containing vertex v_i and its adjacent vertex. The fusion feature represented by vertex v_i, $\hat{h}_i$, can be calculated by the following formula.

$$\hat{h}_i = \sum_{v_j \in N(v_i)} f(v_j, v_i) \times h_j \tag{1}$$

where $\sum_{v_j \in N(v_i)} f(v_j, v_i) = 1$ and $f(v_j, v_i)$ represent the weight that the feature of v_j affects on v_i.

According to Formula (1), we can fuse the features represented by vertices and adjacent vertices directly connected to them. However, in an actual application, the characteristics of v_i are not only affected not only directly by its neighbors but also indirectly by other vertices transferred from its neighbors. According to this idea, Formula (1) can be rewritten as Formula (2) to fuse the features of vertex v_i and all vertices whose distance from v_i is no more than l hops.

$$\hat{h}_i^l = \sum_{v_j \in N(v_i)} f(v_j, v_i) \times \hat{h}_j^{l-1} \tag{2}$$

$\hat{h}_i^l$ represents the fusion feature that contains the feature of vertex v_i and the vertices that are l hops away from v_i, and initially, $\hat{h}_i^0 = h_i$. After information transmission and feature fusion, the feature $\hat{h}_i^l$ of vertex v_i absorbs the features of its neighbors, eliminates the noise in its feature, and then improves the semantic expression ability.

C. FEATURE FUSION WITH RESIDUAL CONNECTION

The previous section discussed the method of fusing the features expressed by vertices whose distances are less than l hops in the text infographic by applying information dissemination. Through information fusion, the vertex contains its own features and those of its surrounding vertices. This increases the diversity of feature representation, eliminates noisy information, and improves the feature expression ability.

However, with the increase in l, the spread of information increases, which leads to the excessive fitting of feature representation, namely, profound assimilation of vertex features, and then the effect decreases rapidly. To solve the problem of overfitting, a residual connection network is applied in the process of information transmission and feature fusion. After each feature fusion, the fused feature is superimposed with the original feature of the corresponding text of the vertex to ensure that the vertex retains its distinctive features based on the features of the surrounding vertices.

According to this idea, we revise Formula (2) as follows so that it superimposes with the original features of the vertex itself after fusing the vertex and surrounding vertex features.

$$\hat{h}_i^l = h_0 + \sum_{v_j \in N(v_i)} f(v_j, v_i) \times \hat{h}_j^{l-1} \tag{3}$$

IV. CLASSIFICATION BASED ON MULTILAYER DAGTs

The previous section discussed the method of information transmission and feature fusion in text infographics, which integrates the features of each vertex in the text infographic with those of associated vertices. For each vertex in a text infographic, the vertices associated with it may belong to different types and have diverse features. Therefore, these associated vertices have an inequitable influence on the labeled classification of the vertex. Next, we use the two-layer attention mechanism to subdivide the relationship of the vertices in the text infographic, distinguish the difference in the connection between related vertices, and label the categories of unlabeled vertices based on this basis.

A. MODEL FRAMEWORK

According to the idea of information transmission and feature fusion based on text infographics, we design a framework model based on a multilayer graph neural network and attention mechanism for multilabel document classification, as shown in figure 2.

In the proposed model framework, we use a l-layer graph neural network to fuse the node features with a distance of l hops around the target node. When feature fusion is carried out in each layer of the graph neural network, class-level and node-level attention networks are applied to distinguish the attention of target nodes to surrounding nodes with different types and features. To avoid the model degradation caused by the increase in network depth, we also use a residual network to superimpose the initial features and fusion features of target nodes to enhance the ability to express its features.

B. TYPE-LEVEL ATTENTION

Given a vertex in a text infographic, we call it the target vertex. Many vertices might connect to the target vertex and may differ in classification. Generally, the features of vertices in the same category are more similar. Based on this, more attention should be given to the vertices belonging to the same category as the target vertex and less attention should be given to the vertices belonging to other categories when fusing the features of the target vertex and its related vertices. In this way, we can enhance the ability of the target vertex to represent its category.

Given a target vertex v_t whose feature vector is h_t, let $N(v_t)$ denote the neighbor vertices of v_t, and let $C_{N(v_t)}$ denote the set of categories of vertices in $N(v_t)$. Then, for any category $c \in C_{N(v_t)}$, we first superposition the feature vectors of all nodes of type c in the adjacent points of v_t and denote it as $h_c (h_c \in \mathcal{R}^m)$ using a method similar to in [31]. Second, we concatenate h_c with h_t to be the feature of v_t about the category c. After that, we obtain the attention score of v_t on c by calculating the similarity between the combined feature vector h_c and the attention vector of type c.

$$score_c = \sigma(\mu_c^T \cdot [W_c h_t \ominus W_c h_c]) \tag{4}$$

where the operator $\ominus$ represents the concatenation operation of the vectors on the left and right sides, $\sigma(\cdot)$ represents an activation function, $W_c \in \mathcal{R}^{m \times m'}$ is a shared linear transformation matrix that is used to transform the m-dimensional input features into m'-dimensional features, and $\mu_c \in \mathcal{R}^{2m'}$ is a shared attention vector that can be obtained by model training.

Finally, the softmax function is used to normalize the attention scores of the target vertex on all types of the surrounding nodes and thus obtain the attention coefficient of the target vertex on every type.

$$\alpha_c = \frac{exp(score_c)}{\sum_{c' \in C_{N(v_t)}} exp(score_{c'})} \tag{5}$$

C. NODE-LEVEL ATTENTION

The previous section discussed how to calculate the attention coefficient of the target node to different types of nodes. This section will continue to discuss how to calculate the attention coefficient of the target node to each adjacent point. The node-attention mechanism differentiates the attentions of the target node to different nodes during feature fusion. In this way, it can capture the vital feature information of adjacent nodes, reduce the noise information brought by secondary nodes, and improve the ability to express its category.

Given a target node v_t and its adjacent nodes $N(v_t)$, we learn the importance of each node $v_i \in N(v_t)$ to v_t by calculating the score of v_t's attention on v_i. After that, the attention scores of the target node to its adjacent nodes are normalized, and the attention coefficient of the target node to each connection node is obtained using the following formula.

$$\beta_{ij} = \frac{exp\left(\mu_v^T \cdot [W_v h_i \ominus W_v h_j]\right)}{\sum_{k \in N(v_i)} exp\left(\mu_v^T \cdot [W_v h_i \ominus W_v h_k]\right)} \tag{6}$$

where $W_v \in \mathcal{R}^{m \times m'}$ is a shared linear transformation matrix and $\mu_v \in \mathcal{R}^{2m'}$ is a shared attention vector.

D. FEATURES FUSION WITH DGAT

Using two-level attention, we distinguish the attention of the target node to different nodes and node types, and fuse the features of the target node with those of the surrounding nodes. According to the relationship between nodes in the text information graph, we can use dual-layer attention mechanism diagram attention network (named DGAT for short) to learn the type attention matrix W_c and node attention matrix W_v, respectively, and then obtain the fused type and node features of the target node. Moreover, the fusion features of the target node can be calculated by concatenating the type and node features using the following Formula (7).

$$h_i = \sigma\Big(\sum_{c \in C_{N(v_i)}} \alpha_c W_c h_c\Big) \oplus \sigma\Big(\sum_{v_j \in N(v_i)} \beta_{ij} W_v h_j\Big) \tag{7}$$

where the operator $\oplus$ represents the superposition operation of the vectors on the left and right sides.

By applying one layer of DGATs, the target node can aggregate the features of the nodes that are $l (l = 1)$ hops away from the target node. For all nodes in the text information graph, we can get their fused features by applying the same method. We denote the set of the fused features as H^l, where $l = 1$.

On the basis that the fusion feature H^l is acquired, we can use another layer of DGATs network to aggregate the fused features of each node in the text information graph again according to formula (7). This is equivalent to that after we have fused the features of the nodes whose distance from the target node is no less than l hops and then aggregate them with the features of nodes whose are $l + 1$ hops away from the target node, as shown in Equation (8).

$$H^{(l+1)} = \sigma\left(H^l\right) \tag{8}$$

Therefore, the further iteration of feature fusion can be completed by superimposing more DGAT networks, which can realize the feature fusion between the target node and the surrounding nodes in a more extensive range.

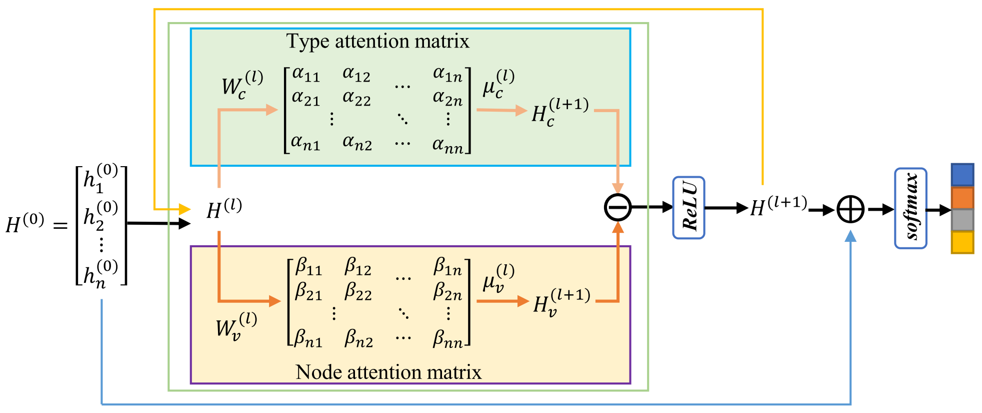

FIGURE 2. Multilayer graph neural network with attention mechanism for text classification.

E. RESIDUAL NETWORK

In the previous subsection, we discussed applying the l layers DGAT to fuse the features of nodes that are no more than l hops away from the target node. As the value of l increases, the target node can aggregate more features of its neighbors and enhance the feature expression ability of the target node. However, as the target node integrates more and more features, the discrimination expressed by the features of the target node decreases gradually. In addition, the vanishing gradient problem becomes more evident as the number of layers of neural networks increase.

When applying multilayer DGAT networks for text classification in our model, we use residual network to solve the problem of the degradation of node feature expression and vanishing gradient, which mainly includes two measures. Firstly, when fusing the fused features obtained from the DGAT network of l^{th} layer again, we apply the activation function ReLU() to eliminate or reduce the weight of unimportant feature information in H^l. Secondly, as shown in Equation (9), we superimpose the original feature H^0 on the final output fusion feature H^{l+1} to enhance the influence of each node's features in classification.

$$H^f = ReLU(H^l) + H^0 \tag{9}$$

where H^f is the fusion feature of the nodes in the text information graph after feature aggregating by applying l layers DGAT networks.

F. COST FUNCTION

After the final fusion feature matrix, H^f, is obtained, we use a softmax layer to map the nodes' features to a probability distribution of their categories.

$$Z = softmax(H^f) \tag{10}$$

According to Equation (10) above, each row vector in Z is the probability distribution of the corresponding node on the category set. We believe that a node's category should coincide with the category with the highest probability.

Given the category set, denoted by C, of the set of documents V, for each document $v(v \in V)$, $v_c = 1$ if the v's category is $c(c \in C)$, otherwise $v_c = 0$. Z_v is the probability distribution of the document v on category set C obtained by our classification model, and Z_v^c is the probability that v belongs to the category c. We use cross-entropy $\sum_{c \in C} v^c \cdot \log(Z_v^c)$ to evaluate the loss caused by classification model for classifying document v. Moreover, we accumulate the loss when classifying all documents in V, and obtain the cost function of our classification model as shown in the following Formula (11).

$$Cost = -\frac{1}{N}\sum_{v \in V}\sum_{c \in C} v^c \cdot \log(Z_v^c) \tag{11}$$

V. EXPERIMENTS

This section first introduces the datasets used in the experiments, the settings of the experimental environment parameters, and the related methods used for performance comparison. Finally, we report the experimental results.

A. DATASETS

We used five reference datasets, including three citation datasets Cora, Citeseer, and PubMed, one protein correlation dataset PPI, and one knowledge graph dataset NELL.

1) CORA

The Cora dataset consists of 2708 papers in the field of machine learning. It has been a popular deep learning graph dataset in recent years. The papers were selected in such a way that in the final corpus every paper cited or was cited by at least one other paper and the papers were classified into seven classes. After stemming and removing stopwords, we were left with a vocabulary of 1433 unique words. All words with a document frequency less than 10 were removed.

2) CITESEER

This is a linked dataset built with permission from the Citeseer web database. Each row represents a scientific paper and each attribute represents an author. In a given row, there is

TABLE 1. Description of the datasets used in the experiments.

Dataset	Nodes	Edges	Features	Classs
Cora	2708	5429	1433	7
Citeseer	3312	4732	3703	6
Pubmed	19717	44338	500	3
PPI	56944	818716	3327	4732
NELL	65755	266144	5414	210

a one for every author associated with that row (i.e., paper) and a zero for every author not associated with that row. The data file is stored in a sparse format and hence we do not expect a giant CSV matrix. The Citeseer dataset consists of 3312 scientific publications classified into one of six classes. The citation network consists of 4732 links. Each publication in the dataset is described by a 0/1-valued word vector indicating the absence or presence of the corresponding word from the dictionary. The dictionary consists of 3703 unique words.

3) PUBMED

The PubMed dataset consists of 19717 scientific publications from the PubMed database pertaining to diabetes classified into one of three classes. The citation network consists of 44338 links. Each publication in the dataset is described by a TF-IDF weighted word vector from a dictionary that consists of 500 unique words.

4) PPI

We make use of a protein-protein interaction (PPI) dataset that consists of graphs corresponding to different human tissues. The dataset contains 20 graphs for training, 2 for validation and 2 for testing. Critically, testing graphs remain completely unobserved during training. To construct the graphs, we used the preprocessed data provided by [32]. The average number of nodes per graph is 2372. Each node has 50 features that are composed of positional gene sets, motif gene sets and immunological signatures. There are 121 labels for each node set taken from Gene Ontology, collected from the Molecular Signatures Database [33]. A node can possess several labels simultaneously.

5) NELL

NELL is a dataset extracted from the knowledge graph introduced in [34]. A knowledge graph is a directed entity graph with a connection relation between nodes. We follow the preprocessing scheme in [35] and assign separate relational nodes R1 and R2 for each entity pair e(e1,r,e2), namely, (e1, r1) and (e2, r2). The entity nodes are represented by sparse feature vectors, and the number of features is expanded by assigning a unique thermal code to each node.

B. PARAMETER SETTINGS

In the experiment, for each dataset, we randomly selected 20 texts from each type of document to form the training sample set together with 500 labeled documents. The remaining documents were used as the testing sample set. We used the three-layer DGAT structure to construct our model, performed more than 200 cycles of training on the training sample set, and tested the correctness of the model on the testing sample set. In model training, all word embeddings were set to 100 dimensions, and document features and type features were set to 512 and 8 dimensions, respectively. The $L2$ regularization term was set to 0.0005 for the Cora and Citeseer datasets, while 0.001 for the PubMed dataset. The learning rate was set to 0.005 for all datasets but 0.01 for PubMed dataset. We report the average of 100 predictions as the model's correctness for each group of experiments.

C. BASELINES

For comparative analysis, we first compare the proposed method with some benchmark methods by running experiments on three citation network datasets and NELL datasets. Benchmark methods for comparison include the manifold regularization (ManiReg) method [37], label propagation (LP) [38], ICA [39], SemiEmb [40], DeepWalk [41], GCN [8], sMGC [42]and GAT-MH [11]. Methods ManiReg, LP, and ICA represent traditional machine learning methods among the above benchmark methods. Methods SemiEmb and DeepWalk represent the methods that combine traditional machine learning methods with deep learning methods. GCN and sMGC are deep learning methods based on graph convolutional neural networks. And GAT-MH is a deep learning method based on a graph attention network.

Moreover, we also compared against the GraphSAGE [32] method by conducting the experiments on the PPI dataset. The GraphSAGE method is presented by Hamilton et al. It is a general inductive framework that leverages node feature information such as text attributes to efficiently generate node embeddings for previously unseen data. In the comparative experiments, our variants of GraphSAGE that use the different aggregator functions are compared, including GraphSAGE-GCN, GraphSAGE-mean, GraphSAGE-LSTM and GraphSAGE-pool.

D. RESULTS

Firstly, we conduct experiments on three citation network datasets and NELL datasets to evaluate the performance of our method. In the experiments, we construct our algorithm model with three-layer DGAT networks, and report the mean classification accuracy (with standard deviation) on the test nodes after 100 runs. As a comparison, the GAT-MH algorithm adopts a two-layer GAT network. The first layer consists of 8 attention heads computing 8 features each, and the second layer is used for classification. Results reported in [8] and [26] were reused for other baseline methods that participated in the comparison experiment.

As shown in Table 2, our method far outperforms traditional machine learning methods and those that combine machine learning with deep learning. The main reason is that traditional machine learning methods mainly use statistical methods to learn the external characteristics of data and cannot obtain the deep correlation relationship contained in

TABLE 2. Accuracy of different methods on four datasets.

Method	Cora	Citeseer	Pubmed	NELL
ManiReg	59.5%	60.1%	70.7%	21.9%
SemiEmb	59.0%	59.6%	71.7%	26.7%
LP	68.0%	45.3%	63.0%	26.8%
DeepWalk	67.2%	43.2%	65.3%	58.5%
ICA	75.1%	69.1%	73.9%	24.1%
GCN	81.5%	70.3%	79.0%	66.0%
sMGC	82.7%	73.3%	80.0%	65.4%
GAT-MH	84.8%	78.4%	79.8%	65.3%
Proposed model	86.9%	82.2%	80.5%	66.3%

the data. Therefore, the classification accuracy of this kind of method is generally low. For those methods that combine machine learning with deep learning, although the application of neural networks can improve the model's generalization ability and computing power, the fundamental logic of the model is still machine learning. Therefore, the accuracy of these methods has been improved, but the extent of improvement is relatively limited.

The deep learning method based on GCN and GAT can not only learn the features of the data itself, but also aggregate the features of the neighbor nodes in the graph to the central node and then learn the internal relationship between the data. Therefore, the accuracy of these methods on four datasets is much higher than that of other methods. Comparatively speaking, the weight of the influence of neighbor nodes on the central node is determined by the degree of the node in the method based on GCN but by the correlation of features of neighboring nodes in GAT-based methods. Thus, the method based on GAT can achieve better performance because it can mine the deeper relationship between the data.

DGATs network structure, in which each layer of DGAT is composed of two GAT networks. DGAT network structure is equivalent to a graph attention network with two attention heads, which can simultaneously learn the attention coefficients of the data on categories and connection relationships. At the same time, multi-layer DGATs can achieve high-dimensional node feature aggregation, which can improve the feature expression ability of the model. Although GAT-MH uses a graph attention network with eight attention heads and can learn more attention coefficients on features, the feature dimensions that positively impact classification results are still limited in most cases. In addition, GAT-MH only aggregates the features of nodes directly connected to the central node, which may lose some vital feature information. The experimental results on the NELL dataset show that compared with the GAT-MH method, our method's accuracy is improved by 2.48%, 4.84%, 0.9%, and 1.53%, respectively.

Secondly, we also conducted several experiments on the protein-protein interaction (PPI) dataset to compare the proposed model with GraphSAGE and other methods. In the experiment, the results of Random and four GraphSAGE models were obtained from the literature [35]. The structure of the GAT-MH model is a two-layer GAT network containing four attention heads, respectively.

TABLE 3. Accuracy of different methods on PPI dataset.

Method	PPI
Random	39.6%
GraphSAGE-GCN	50.0%
GraphSAGE-mean	59.8%
GraphSAGE-LSTM	61.2%
GraphSAGE-pool	60.0%
GAT-MH	97.3%
Proposed model	97.8%

TABLE 4. Accuracy of proposed method on four datasets as *l* varies.

Layers	Cora	Citeseer	Pubmed	NELL
1	87.06%	76.33%	80.50%	66.30%
2	86.51%	78.11%	79.83%	66.41%
3	87.98%	81.66%	81.72%	66.86%
4	87.43%	80.50%	81.63%	66.23%
5	87.24%	79.63%	81.37%	65.87%
6	86.69%	79.52%	80.52%	66.39%
7	87.06%	79.30%	80.61%	66.29%

The GraphSAGE model constructs the graph structure of the dataset based on the data's geometric features, then performs feature aggregation for the local subgraphs of each node in the graph, and proposes four types of aggregation functions. The GraphSGAE method is similar to the GAT-based methods in aggregating node features using data correlation. Although the GraphSAGE model presents four different aggregation functions to aggregate data features, these four aggregation functions mainly aggregate data mathematical features and lack aggregation of data semantic features. As shown in Table 3, the accuracy of GAT-MH and our proposed model is far better than that of the GraphSAGE model. The GAT-MH model uses a two-layer neural network, which can well fuse the features of nodes with a distance of no more than two hops from the surrounding nodes. Compared with our model, GAT-MH does not solve the problem of gradient disappearance in multi-layer neural networks. Therefore, GAT-MH cannot use a deeper network model to learn deeper features. However, our model applies residual networks to improve the aggregation ability of multi-layer neural networks. Therefore our model achieves better performance than the GAT-MH method.

In addition, we also performed multiple experiments on four citation network datasets. In the experiments, we analyzed the influence of the layer number of DGAT networks on the model correctness by changing the layer number of DGAT networks from 1 to 7.

As shown in Table 4, the experimental results on the four citation datasets show that the model has the highest accuracy when the layer number of DGAT networks is set to 3. These results indicate that when the central node aggregates the features of nodes whose surrounding distance is no more than three hops, the fusion features of the central node have the most robust expression ability. With the increase of the number of layers in DGAT networks, although the central node integrates more characteristics of nodes, the correlation of nodes will be significantly reduced as the distance between two nodes increases. The central node aggregates the features of nodes far away from the central node, which may introduce obscure features and subsequently reduce the model's accuracy.

VI. CONCLUSION

According to the intrinsic correlation of text data, this paper constructs a dual-layer attention mechanism graph neural network (DGAT) to fuse the features of target and local neighborhood nodes. In addition, we can also use the residual network to superimpose multiple DGATs to control the range of local neighborhood nodes participating in feature fusion. This model can aggregate the features of the target node, the features of its neighborhood nodes, and their internal relations, which can effectively improve the ability of node feature expression and classification accuracy. Moreover, the model only fuses the features of nodes in the local domain, dramatically reducing the computational complexity. We implemented the model and conducted extensive experiments on five benchmark datasets. Experimental results show that the proposed model has obvious advantages over other models and methods.

In the future, we will continue this work in two ways. First, we will further modify the model to improve the model's accuracy and generalization ability as well as reduce the computation complexity. Secondly, we will apply our model to solve specific application problems, verify and optimize our model through practice.

REFERENCES

[1] N. Ghamrawi and A. Mccallum, "Collective multi-label text classification," in *Proc. ACM Int. Conf. Inf. Knowl. Manage. (CIKM)*, Bremen, Germany, Nov. 2005, pp. 195–200.

[2] C. C. Aggarwal and C. Zhai, "A survey of text classification algorithms," in *Mining Text Data*, C. C. Aggarwal and C. X. Zhai, Eds. New York, NY, USA: Springer, 2012, ch. 6, pp. 163–222.

[3] M. Lan, C.-L. Tan, H.-B. Low, and S.-Y. Sung, "A comprehensive comparative study on term weighting schemes for text categorization with support vector machines," in *Proc. Special Interest Tracks Posters 14th Int. Conf. World Wide Web (WWW)*, 2005, pp. 1032–1033.

[4] A. M. Kibriya, E. Frank, B. Pfahringer, and G. Holmes, "Multinomial naive Bayes for text categorization revisited," in *Proc. Australas. Joint Conf. Artif. Intell.*, Dec. 2004, pp. 488–499.

[5] N. Suguna and K. Thanushkodi, "An improved k-nearest neighbor classification using genetic algorithm," *Int. J. Comput. Sci. Issues*, vol. 7, no. 2, pp. 18–21, 2010.

[6] F. Harrag, E. El-Qawasmeh, and P. Pichappan, "Improving Arabic text categorization using decision trees," in *Proc. 1st Int. Conf. Netw. Digit. Technol.*, Jul. 2009, pp. 110–115.

[7] A. Genkin, D. D. Lewis, and D. Madigan, "Large-scale Bayesian logistic regression for text categorization," *Technometrics*, vol. 49, no. 3, pp. 291–304, Aug. 2007.

[8] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. 5th Int. Conf. Learn. Represent.*, Apr. 2017, pp. 1–14.

[9] L. Yao, C. Mao, and Y. Luo, "Graph convolutional networks for text classification," in *Proc. 33th AAAI Conf. Artif. Intell.*, Jan. 2019, pp. 7370–7377.

[10] F. Wu, T. Zhang, J. Souza, C. Fifty, T. Yu, and K. Weinberger, "Simplifying graph convolutional networks," in *Proc. 36th Int. Conf. Mach. Learn.*, Long Beach, CA, USA, Jan. 2019, pp. 6861–6871.

[11] P. Velikovi, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *Proc. 6th Int. Conf. Learn. Represent.*, Vancouver, BC, Canada, Apr. 2018, pp. 1–12.

[12] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, 1989.

[13] M. Gori, G. Monfardini, and F. Scarselli, "A new model for learning in graph domains," in *Proc. IJCNN*, Montreal, QC, Canada, Jul. 2005, pp. 729–734.

[14] P. Wang, B. Xu, J. Xu, G. Tian, C. Liu, and H. Hao, "Semantic expansion using word embedding clustering and convolutional neural network for improving short text classification," *Neurocomputing*, vol. 174, pp. 806–814, Jan. 2016.

[15] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," in *Proc. 52nd Annu. Meeting Assoc. for Comput. Linguistics (Long Papers)*, vol. 1, 2014, pp. 655–665.

[16] R. Johnson and T. Zhang, "Deep pyramid convolutional neural networks for text categorization," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics (Long Papers)*, vol. 1, 2017, pp. 562–570.

[17] Y. Meng, J. Shen, C. Zhang, and J. Han, "Weakly-supervised neural text classification," in *Proc. 27th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2018, pp. 983–992.

[18] C. Yin, J. Xiang, H. Zhang, J. Wang, Z. Yin, and J.-U. Kim, "A new SVM method for short text classification based on semi-supervised learning," in *Proc. 4th Int. Conf. Adv. Inf. Technol. Sensor Appl. (AITS)*, Aug. 2015, pp. 100–103.

[19] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "Computational capabilities of graph neural networks," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 81–102, Jan. 2009.

[20] E. A. Smirnov, D. M. Timoshenko, and S. N. Andrianov, "Comparison of regularization methods for ImageNet classification with deep convolutional neural networks," *AASRI Proc.*, vol. 6, no. 1, pp. 89–94, 2014.

[21] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. 53th Annu. Meeting Assoc. Comput. Linguistics*, Beijing, China, Jul. 2015, pp. 1–14.

[22] K. Zhu and M. Cao, "A semantic subgraphs based link prediction method for heterogeneous social networks with graph attention networks," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2020, pp. 1–8.

[23] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[24] J. Bruna, W. Zaremba, A. Szlam, and Y. Lecun, "Spectral networks and locally connected networks on graphs," in *Proc. 52th Annu. Meeting Assoc. Comput. Linguistics*, Baltimore, MD, USA, Jun. 2014, pp. 1–14.

[25] M. Zitnik, M. Agrawal, and J. Leskovec, "Modeling polypharmacy side effects with graph convolutional networks," *Bioinformatics*, vol. 34, no. 13, pp. i457–i466, Jul. 2018.

[26] F. Monti, D. Boscaini, J. Masci, E. Rodola, J. Svoboda, and M. M. Bronstein, "Geometric deep learning on graphs and manifolds using mixture model CNNs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5425–5434.

[27] M. Fey, J. E. Lenssen, F. Weichert, and H. Ḿuller, "SplineCNN: Fast geometric deep learning with continuous B-Spline kernels," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 869–877.

[28] M. Henaff, J. Bruna, and Y. LeCun, "Deep convolutional networks on graph-structured data," 2015, *arXiv:1506.05163*.

[29] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. 30th Int. Conf. Neural Inf. Process. Syst.*, Barcelona, Spain, Dec. 2016, pp. 3844–3852.

[30] H. Peng, J. Li, Y. He, Y. Liu, M. Bao, L. Wang, Y. Song, and Q. Yang, "Large-scale hierarchical text classification with recursively regularized deep graph-CNN," in *Proc. World Wide Web Conf. World Wide Web (WWW)*, 2018, pp. 1063–1072.

[31] 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.

[32] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, Long Beach, CA, USA, Dec. 2017, pp. 1025–1035.

[33] A. Subramanian, P. Tamayo, V. K. Mootha, S. Mukherjee, B. L. Ebert, M. A. Gillette, A. Paulovich, S. L. Pomeroy, T. R. Golub, and E. S. Lander, "Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles," *Proc. Nat. Acad. Sci. USA*, vol. 102, no. 43, pp. 15545–15550, Aug. 2005.

[34] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. Hruschka, and T. M. Mitchell, "Toward an architecture for never-ending language learning," in *Proc. 24th AAAI Conf. Artif. Intell.*, Atlanta, GA, USA, Jul. 2010, pp. 1306–1313.

[35] Z. Yang, W. W. Cohen, and R. Salakhutdinov, "Revisiting semi-supervised learning with graph embeddings," in *Proc. 33rd Int. Conf. Int. Conf. Mach. Learn.*, New York, NY, USA, Jun. 2016, pp. 40–48.

[36] C. Subakan, M. Ravanelli, S. Cornell, M. Bronzi, and J. Zhong, "Attention is all you need in speech separation," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 21–25.

[37] M. Belkin, P. Niyogi, and V. Sindhwani, "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples," *J. Mach. Learn. Res.*, vol. 7, pp. 2399–2434, Jan. 2006.

[38] X. Zhu, Z. Ghahramani, and J. D. Lafferty, "Semi-supervised learning using Gaussian fields and harmonic functions," in *Proc. 20th Int. Conf. Mach. Learn. (ICML)*, Washington, DC, USA, Aug. 2003, pp. 912–919.

[39] L. Getoor, "Link-based classification," in *Advanced Methods for Knowledge Discovery From Complex Data*. London, U.K.: Springer, 2005, ch. 7, pp. 189–207.

[40] J. Weston, F. Ratle, and R. Collobert, "Deep learning via semi-supervised embedding," in *Proc. 25th Int. Conf. Mach. Learn. (ICML)*, 2008, pp. 1168–1175.

[41] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online learning of social representations," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2014, pp. 701–710.

[42] J. Zhang, B. Hui, P.-W. Harn, M.-T. Sun, and W.-S. Ku, "MGC: A complex-valued graph convolutional network for directed graphs," 2021, *arXiv:2110.07570*.

HUI CHEN (Member, IEEE) received the B.S. degree in vehicle engineering from Tongji University, China, in 1999, and the M.S. degree in computer application technology and the Ph.D. degree in computing software and theory from the Huazhong University of Science and Technology, Wuhan, China, in 2004 and 2008, respectively.

He is currently a Full Professor at Foshan University, Foshan, China. His current research interests include data mining and big data analysis.

JIAN HUANG received the B.S. degree in electronics and information technology from Nanchang University, Nanchang, China, in 1999, and the M.S. degree in circuit and system and the Ph.D. degree in communication and information system from Wuhan University, Wuhan, China, in 2004 and 2008, respectively.

He is currently an Associate Professor at the Jiangxi University of Finance and Economics, Nanchang. His current research interests include embedded systems, machine learning, and big data analytics.

NANA TAO is currently pursuing the master's degree with the Jiangxi University of Finance and Economics, majoring in electronic and communication engineering. Her current research interests include machine learning and big data analytics.

JIJIE HUANG is currently pursuing the master's degree with Foshan University, majoring in software engineering. His current research interests include big data analysis and deep learning.

JING WANG (Member, IEEE) received the B.S. degree in computer science from Wuhan University (WHU), China, in 2003, the M.S. degree in software engineering from the Huazhong University of Science and Technology, China, in 2006, and the Ph.D. degree from the State Key Laboratory of Software Engineering (SKLSE), WHU, in 2011.

He is currently an Associate Professor at the School of Software and Internet of Things Engineering, Jiangxi University of Finance and Economics (JXUFE). His current research interests include evolutionary computing and parallel computing.