IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 26 January 2023, accepted 15 February 2023, date of publication 21 February 2023, date of current version 24 February 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3246525

APPLIED RESEARCH

Enhancement Economic System Based-Graph Neural Network in Stock Classification

YAOQUN XU[1,2] AND YUHANG ZHANG[1]
[1]School of Computer and Information Engineering, Harbin University of Commerce, Harbin 150028, China
[2]Institute of System Engineering, Harbin University of Commerce, Harbin 150028, China

Corresponding author: Yaoqun Xu (xuyq@hrbcu.edu.cn)

This work was supported by the Natural Science Foundation of Heilongjiang Province under Grant LH2021F035.

ABSTRACT As a result of the integration of the stock industry into the entire international economic system, stock companies publish hundreds of prospectuses every second. The ability to quickly and accurately classify the companies and categories to which these data belong, as well as improve the performance of the economic system, has become the key to unlocking its corresponding value at this stage. In order to solve this problem, graph neural network techniques are used to accomplish the classification of stocks in the science and technology version, thus indirectly alleviating the enormous pressure on the economic system. In this study, to complete semi-supervised classification, we propose the PA-GCN model, which takes the lead in graph attention calculation of stock nodes and introduces the Elu activation function. Specifically, in this study, we constructed a stock dataset and implemented a dropout layer to prevent overfitting. The final results on the stock dataset and the publicly accessible Cora dataset demonstrate that this strategy may effectively improve the economic system and that the model has good performance. In the two dataset tests, classification accuracy can be attained at 81.69% and 81.2%, respectively. It also demonstrates the method's viability for classifying stock nodes.

INDEX TERMS Graph neural network, node classification, graph convolutional neural network, graph attention mechanism.

I. INTRODUCTION

With the rapid integration of the Internet and the financial sector, the stock market entered people's production and daily lives, and the classification method based on deep learning and graph neural network began to be applied to a variety of emerging fields. In this study, we use graph neural networks and other techniques to complete the classification of stock nodes in the science and technology version using data from prospectuses. This is an excellent and intriguing piece of research that not only aims to improve the economic system, but also to extend the domain of graph neural networks and finish the large-scale integration of graph neural networks with the economic domain. In addition, there are few relevant studies on the classification of corporate nodes using the prospectus as the primary data source, so the classification of stock nodes in the Scientific innovation version represents a novel concept and an original effort [1].

Compared to other standard data necessary for graph neural network node classification, the data in the prospectus is very intricate and structurally diversified. The Internet-crawled prospectus has a relatively chaotic structure and complex content sources, among other characteristics. Each company's prospectus has keyword extraction similarities, which poses a challenge to the experimental results. Therefore, data cleansing should be performed after keyword extraction to eliminate the irrelevant portion of the data that has a significant impact on the experimental outcomes [2]. In addition, the low cost of selecting a small amount of training data can result in a high classification accuracy, which is the focus of this paper.

In recent years, the number of publicly traded corporations has expanded, resulting in more appropriate and accurate statistics. Graph neural network is an emerging research direction in the field of artificial intelligence, and its

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

emergence compensates for the severe lack of graph learning technology [3].

In this paper, the company's stock classes are classified using the prospectus's keywords as a database. The crawler technology will be used to crawl the prospectuses of the listed companies in the scientific innovation version of Eastmoney.com, which will be divided into eight categories based on the industry category. Using technology for natural language processing to extract the prospectus's keywords. After data extraction is complete, keywords with a high repetition rate and no useful classification are removed. To complete the construction of the dataset, logic and operation are used to calculate the weight relationship between companies based on keyword information. A PA-GCN model enhancement completes the semi-supervised classification of stock nodes. This paper examines the relevant literature on node classification and proposes a semi-supervised stock node classification model based on graph convolutional neural network. The semi-supervised classification of stock nodes using a graph convolutional neural network is completed by self-constructing the required dataset of stock classification. In conclusion, this paper's contribution is as follows:

·In this study, the graph attention mechanism takes the lead in weighting the node information before the data enters the graph convolution layer and replaces the default activation function to improve the classification accuracy of the model. In order to prevent the model from becoming overfit during training, the Dropout layer is added at the end of the model. The experimental results indicate that this model has some advantages over other models.

· Natural language processing and logic and operation techniques were used to create a stock classification dataset suitable for graph convolutional neural network.

· Empowering economic systems to complete graph neural network domain expansion.

II. RELATED WORK

In the Sci-innovation version of Eastmoney.com, the stock classification of listed companies is essentially a nodal classification of stock companies. With the rise of deep learning and the introduction of graph neural networks, node-level classification based on graph neural networks has gradually become one of the research community's primary focuses. In the interim, inspired by the success of deep learning on grid-structured data, Wu et al. [4] propose graph neural network models for learning powerful node-level or graph-level representations. However, there are limitations, so we propose a Demo-Net-specific neural network that can recognize 1-hop neighborhood structure recursively. It is also applied to multiple datasets, with positive results for node classification. Wang et al. [5] proposed a semi-supervised classification of nodes in a Markov random walk graph neural network and integrated the graph neural network with Markov random walk and graph neural network. Through the final experimental comparison, the method proposed in this paper can effectively improve classification accuracy and play an important role in encouraging future experimentation. Xu et al. [6] achieved remarkable success in the classification of graph-based semi-supervised nodes using a graph neural network, and they proposed a graph neural network based on label consistency, which utilized node pairs without connection but with the same label to expand the acceptance domain of nodes in GNN. Multiple datasets demonstrate that the proposed model is significantly superior to the conventional graph neural network model. Based on the remarkable success of graph neural network in processing graph structure data, Yuan et al. [7] designed GNN with strong representation capability in response to the node classification problem. However, the depth model has a problem with overfitting, so a novel concept of aggregating more useful information based on multiple views without depth structure is proposed, and a large number of node classification experiments are conducted on six public datasets, demonstrating the superiority of the proposed model over the most recent methods. Huang et al. [8] improved classification of citation networks and obtained better results. The classification of points by a graph neural network has been completed successfully. Qiang et al. [9] classified bidding documents using a graph neural network, made full use of relevant knowledge of node classification, adopted a good graph neural network model, and obtained the expected result. This experiment is an invaluable resource for future studies in this field. Dabhi and Parmar [10] proposed a graph regularization neural network for node classification and a model using Ncl-Nodenet to solve the citation graph node classification task. The relevance of modification to the task will be discussed. Comparing the results of this experiment with the current state of technology, the superior performance of NodeNet is investigated. Wu et al. [11] proposed a supplementary training method to improve the classification of semi-supervised graph neural network nodes. You can extract and generate labels for pairs of nodes on this page. By noting the supervision of pairwise nodes, it is possible to compel the predicted tags to conform to the observed pairwise relationships and provide valuable information to enhance performance. The evaluation results of a large number of experiments indicate that this experiment's framework is superior. Wang et al. [12] proposed modeling the output-output relationship with explicit paired factors and the GNN backbone was used to model the input-output relationship. In order to strike a balance between the model's complexity and expressiveness, the paired factors have a shared component and a unique scaling factor for each edge. Experiments on various data sets demonstrate that the proposed model can effectively enhance the performance of semi-supervised node classification. Zhang et al. [13] accurately predicted and classified unlabeled nodes in a graph in order to extract features from graph data more effectively. GCN was used to extract data feature information from the Cora dataset, and the extracted data was then integrated and categorized. The experiment's feasibility was demonstrated by modifying and debugging the code's learning rate, attenuation weight, and training times. Chen and Guo [14] proposed

a simple complex- heterogeneous graph attention neural network method,

SC-GANN, in order to better learn higher-order information and heterogeneous information in networks. The proposed method has improved macro-F1, micro-F1, precision, and recall on all three datasets compared to GCN and HAN. The results demonstrate that this method can effectively learn higher-order information and heterogeneous information in a network and improve node classification precision.

In this study, the deep learning classification algorithm and graph neural network (GNN)-related technology are utilized to classify node of Sci-Tech. Graph neural network is an end-to-end learning model based on graph data that combines graph data and deep learning effectively. The concept of graph neural network was first proposed by Gori et al. [15], who devised a model for processing graph structure and graph data by referencing the neural network research accomplishments. Scarselli et al. [16] elaborated on the aforementioned graph model, after which a large number of new graph neural network models and application fields were proposed in a widespread fashion. Bruna et al. [17] were motivated to introduce convolution into graph neural networks for the first time, which has garnered significant industry interest. On the basis of related technologies such as relational induction and deep learning, Battaglia et al. [18] proposed the graph network concept oriented to relational reasoning and combined it with deep learning technology to solve the problem that deep learning cannot be applied to relational reasoning. Zhang et al. [19] reviewed semi-supervised and unsupervised perspectives on graph structure-based deep learning technologies. The development of graph neural networks has accelerated in recent years. In light of the existing issues with the original neural network, researchers have also proposed a number of ideal solutions. With more in-depth research and exploration of the field of graph neural network, the graph neural network application field is destined to expand significantly. There are less relevant papers investigating these concerns in past research. Comparing the graph structure according to the related similar articles, it can be seen that the node classification of the completed graph neural network cannot improve the classification accuracy on the original graph structure without changing the number of layers of convolutional layers. In order to resolve this issue, a novel concept was developed. The graph attention is calculated first before the data enters the graph convolution layer to enhance the accuracy of the weights assigned to the nodes when forming the graph structure, which indirectly improves the accuracy of the data and thus completes the improvement of the classification accuracy in the overall structure. In this study, the semi-supervised node classification is done by using the self-built dataset and the public Cora dataset respectively in the PA-GCN model.

III. ELEVANT MODELS AND TECHNIQUES

A. GRAPH CONVOLUTIONAL NEURAL NETWORK

Currently, the expansion of convolutional neural network usage is closely related to the development of deep learning technology. Convolutional neural networks and graph convolutional neural networks share similar properties. It is also clear that graph neural network is an important subfield of neural network, and that the majority of models in use today are derived from neural network. Graph convolutional neural network (GCN) is a model co-evolved from spectral convolutional neural network (SCNN) and Chebshev network (ChebNet) [20].

The earliest application of convolutional neural network to graph data is the spectral convolutional neural network. The model defines the convolution layer of a graph neural network as a filter and implements the graph convolution operation of node information and convolution kernel in spectral domain based on the convolution theorem. However, this method takes into account that the node domain in which the graph resides does not satisfy translation invariance to a large extent, making it challenging to define the graph convolution on the spatial domain. The graph convolution theorem is required for this definition of graph convolution. The graph data is transformed from the spatial domain to the spectral domain using the graph Fourier transform, the convolution calculation is performed, and then the graph data is returned to the spatial domain using the inverse Fourier transform.

According to the graph convolution theorem, the product of spatial domain is converted into the product of spectral domain, as shown in Equation (1).

$$F[f_1(t) * f_2(t)] = F_1(w)F_2(w) \tag{1}$$

where $f(t)$ represents the spatial domain signal, $F(w)$ represents the frequency domain signal, $*$ represents the convolution, and F represents the Fourier transform. $f_1(t)$ represents the spatial input signal, $f_2(t)$ represents the spatial convolution kernel, $F_1(w)$ represents the frequency domain input signal, and $F_2(w)$ represents the frequency domain convolution kernel.

Equation (1) can be transformed into Equation (2) when the inverse Fourier transform is performed to return to the airspace.

$$f_1(t) * f_2(t) = F^{-1}[F_1(w)F_2(w)] \tag{2}$$

The discrete classical Fourier transform formula is shown in Equation (3).

$$F(\omega) = F[f(t)] = \sum_{t=1}^{n} f(t)e^{-i\frac{2\pi}{n}\omega t} \tag{3}$$

In the above equation, $f(t)$ represents the signal, $e^{-i\frac{2\pi}{n}\omega t}$ represents the basis function, and $F(\omega)$ is the Fourier coefficient.

The classical inverse Fourier transform formula is shown in Equation (4).

$$f(t) = F^{-1}[F(\omega)] = \frac{1}{n}\sum_{\omega=1}^{n} F(\omega)e^{i\frac{2\pi}{n}\omega t} \tag{4}$$

Nevertheless, SCNN has quite a few disadvantages:

(1) High computational complexity makes it easy to overfit when the number of nodes is large;

(2) Compute the feature decomposition of the Laplacian matrix is time-consuming and expensive;
(3) SCNN is fully spatially connected, as opposed to locally connected, and has a vast number of parameters.

In accordance with the method of spectral convolutional neural networks, the application cost of the model is high,

the steps are complicated, and it lacks the correlation property. Therefore, He et al. [21] began to see the Chebyshev inequality differently. ChebNet is introduction completely resolves the existing issues and eliminates SCNN is shortcomings.

ChebNet replaces the convolution kernel of the spectral domain with Chebyshev polynomials, the graph convolution of which is illustrated by Equation (5).

$$g_{\theta'} \star x \approx \sum_{k=0}^{K} \theta'_k T_k(\tilde{L})x \tag{5}$$

In the above equation, $\tilde{L}$ represents the maximum characteristic value of L; θ'_k is Chebyshev coefficient vector; g_θ is the convolution kernel of the spectral domain, and when approximated by a Chebyshev polynomial interpolation, the Laplacian matrix L can be used directly.

ChebNet offers the following benefits:

(1) There is only K+1 learnable parameter and relatively few parameters in the convolution kernel;
(2) After using Chebyshev polynomials to replace the convolution kernel in the spectral domain, the Laplace matrix L is no longer needed for the eigen decomposition, and the transformation can be done directly using L, reducing the computational complexity;
(3) The convolution kernel is localized, and K is the perceptual field radius of the convolution kernel, i.e., the Kth order nearest neighbor nodes of the central node is used as neighbor nodes. K = 1 is a weighted summation of the features of the first-order neighbors of each node. K=2 is the weighted summation of the second-order neighboring features of the node. The intuitive schematic is shown in Figures 1 and 2.

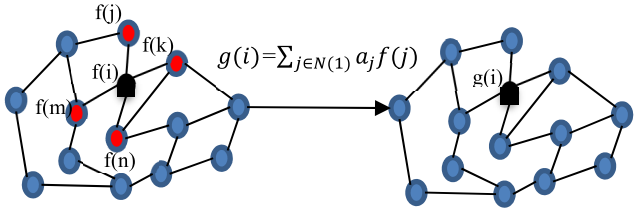

FIGURE 1. An intuitive diagram of k=1.

For the record, the preceding diagram depicts a single node. In the actual ChebNet, the weighted sum of the graph above is applied to each vertex in the convolution graph. The black node is the compute node that is selected, the red node is the first-order or second-order node that is selected, and the blue node is not selected.

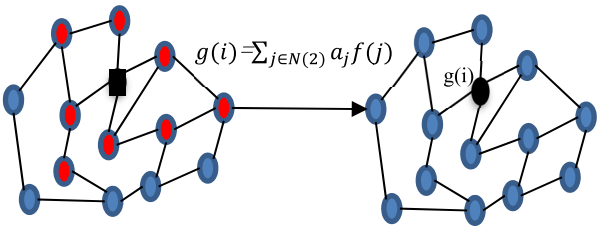

FIGURE 2. An intuitive diagram of k=2.

B. PA-GCN MODEL CLASSIFICATION

Since the majority of ideas for employing attention mechanisms are intertwined with models of graph convolutional neural networks, a novel concept is proposed. When the graph structure information of nodes is not included in the convolution calculation, classification results are obtained by calculating the corresponding weight distribution of stock nodes. In this paper, we use a self-proposed graph convolutional neural network PA-GCN, which integrates attention mechanism calculation and Elu activation function, to perform semi-supervised classification of stock nodes. A small number of label samples are used to train the model, a small number of label nodes are used to train the model, and a large number of non-training nodes are classified.

1) MODEL STRUCTURE

Input layer, graph attention layer, graph convolution layer, and output layer make up the majority of the enhanced PA-GCN model. Figure 3 depicts the structure of the graph convolution model overall.

2) INPUT LAYER

Input layer: The graph structure information consisting of node data and edge data is converted into an adjacency matrix and a degree matrix before being input into the graph attention layer to first complete weight calculation and then input information to finish training.

3) GRAPH ATTENTION LAYER

Graph attention layer: Calculate the attention coefficient based on the obtained calculated data. Calculate the coefficients of similarity between vertex I and its neighbors one by one. As indicated by Equation (6).

$$e_{ij} = a([Wh_i||Wh_j]), j \in N_i \tag{6}$$

In the above equation, h can be represented as a series of embedded vectors in the above equation. W means to increase the dimension of the vertex's features;|| for the vertex's i, j characteristics after splicing; $a()$ represents the mapping of the concatenated high-dimensional feature to a real number. e_{ij} is the attention coefficient in its raw form.

The attention coefficient is then normalized to yield the final attention coefficienta_{ij}. As indicated by Equation (7).

$$a_{ij} = \frac{\exp(LeakyReLU(e_{ij}))}{\sum_{k\in N_i} \exp(LeakyReLU(e_{ik}))} \tag{7}$$

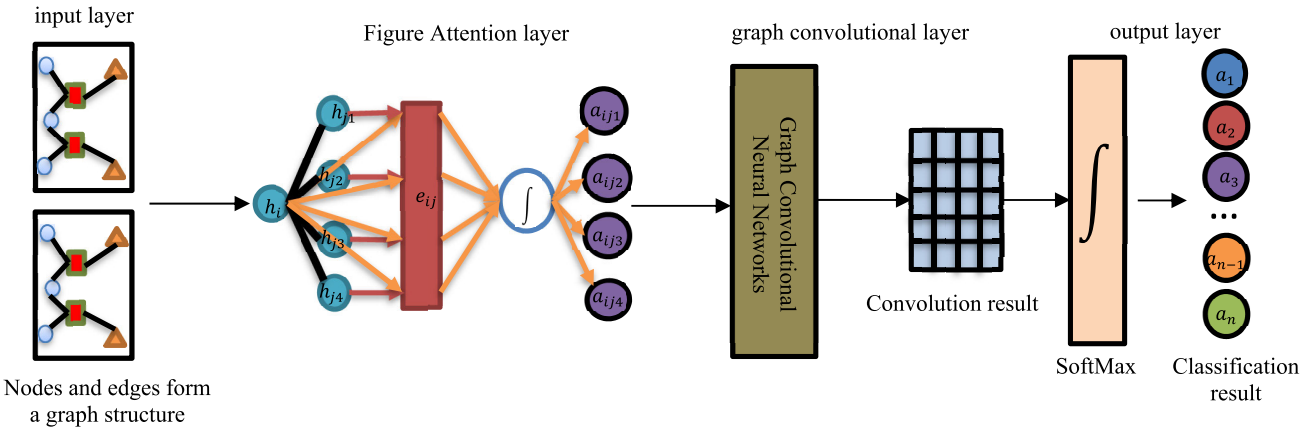

FIGURE 3. Classification structure of graph convolutional neural network.

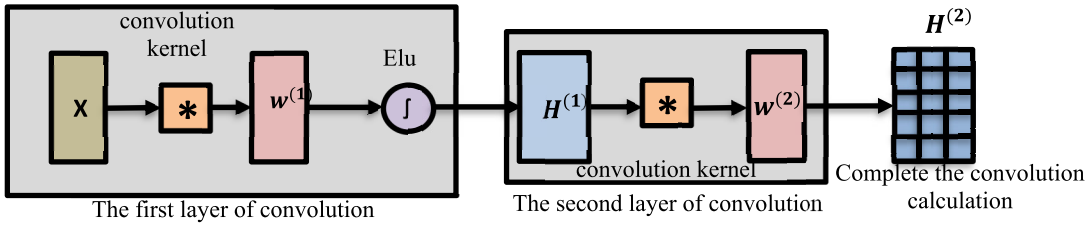

FIGURE 4. Schematic diagram of the GCN module.

Finally, the features are then weighted and summed in accordance with the attention coefficient. As indicated by Equation (8).

$$h_i^{l+1} = \sigma(\sum_{j \in N_i} a_{ij} W h_j) \tag{8}$$

In the above equation, W is the weight matrix of feature multiplication; a represents the final attention coefficient calculated above; σ means nonlinear activation function; $j \in N_i$ means to traverse all nodes adjacent to i.

4) GRAPH CONVOLUTION LAYERS

Graph Convolution Layer: Before feeding data to the graph attention network, a two-layer convolutional GCN model is used to train the model. Equations (9) and (10) depict the two-layer graph convolutional network.

$$H^{(1)} = Elu(\hat{A}XW^{(1)}) \tag{9}$$

$$H^{(2)} = \hat{A}H^{(1)}W^{(2)} \tag{10}$$

In the above equation, $H^{(1)}$ represents the output of the first layer as well as the input of the second layer; $H^{(2)}$ indicates the output of the second layer; *Elu* denotes the activation function; X denoted as the input to the first layer of the neural network. W denotes the weights of the graph neural network. $W^{(1)}$ denotes the weights of the first layer of the neural network. $W^{(2)}$ denotes the weights of the second layer of the neural network. $\hat{A}$ is a single layer convolution operation that can be interpreted as a convolution kernel, as shown in Equation (11).

$$\hat{A} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}} \tag{11}$$

Figure 4 illustrates a diagram of the GCN module.

A graph convolutional neural network maps the data G = (V, E) of the original graph structure into a new vector space $f^G \rightarrow f^*$ [22]. using a single-layer forward-propagating graph convolutional neural network as an example, the features of the resulting layer i neural network can be represented by ω_i. In calculating each node v_i in the graph, the output H^{l+1} of any neural network corresponding to each layer can be represented by a non-linear function $f(\cdot, \cdot)$ of the form $H^{l+1} = f(H^l, A)$, where A is the feature adjacency matrix. The graphical convolutional network structure is implemented by the nonlinear activation function $ReLU = \sigma(\cdot)$, and the propagation rules for the layering are depicted in the equation (10).

$$f(H^l, A) = \sigma(\hat{A}^{-1/2}\hat{A}\hat{D}^{-1/2}H^l W^l) \tag{12}$$

In the above equation, $\hat{A} = A + I$ represents the adjacency matrix in the graph structure, I represent the unitary array, $\hat{D} = \sum_j \hat{A}_{ij}$ represents the diagonal array of matrix $\hat{A}$, and W^l represents the weight matrix of the corresponding lth layer of the convolutional neural network [23].

Using the aforementioned hierarchical propagation rules, the shared parameters are sequentially introduced into the

graph structure, allowing each node to perceive that the propagation's breadth increases with the number of layers, thereby gaining more knowledge about its neighboring nodes [24]. By applying a two-layer graph convolution operation to the weight matrix calculated after fusing the attention mechanism at the input, the convolution result is finally output in vector form at the output layer.

5) OUTPUT LAYERS

Output layer: The SoftMax function classifies the results of the second training layer to generate the final category labels. The output label category expressions are as follows: equation (13).

$$Z = softmax(H^{(2)}) \tag{13}$$

In the preceding equation, $H^{(2)}$ represents the result of the convolution operation as the input layer input; Z represents the final result of the classification.

C. JIEBA WORD SPLITTING TECHNIQUES

A Chinese utterance is a unique sequence of characters consisting of a combination of words and phrases, and the same character can have different meanings in different sentences. In order for the computer to accurately comprehend the text and extract the main keywords, it must first perform a word separation operation to divide the text according to certain rules; hence, the development of jieba word separation [25]. Jieba is a popular third-party Python library for Chinese word splitting. Python provides an interface for jieba is word splitting functionality. The jieba word separation algorithm is simple, accurate, suitable for Chinese word separation, supports three-word separation modes, and can precisely calculate the weight occupied of keywords, making it a potent algorithm [26], [27]. Figure 5 depicts the functional structure of the jieba split.

However, accessing the necessary text data from an open Internet platform is difficult. Due to the complexity and diversity of the required data sources, the process of extracting keywords and performing word frequency analysis using jieba splitting still produces a substantial amount of incomplete and nonconforming data, noise, etc. This is an objective aspect that cannot be avoided. Therefore, it is necessary to manually correct the filtered feature words to improve the accuracy of the word separation and prevent any impact on the required data. Extraction and correction procedures for this experiment are described in the following section.

D. LOGIC AND ARITHMETIC TECHNIQUES

With, or, and not are the three types of logical operations. The symbol for the logical sum operation is "&", and the rule is that if "0" signifies false and "1" signifies true, then the result can only be 1 if both numbers are true, i.e., if the data are both 1 [28]. If the first operand in a logical sum operation on any matrix is 0, the result cannot be 1 regardless of the second operand's value.

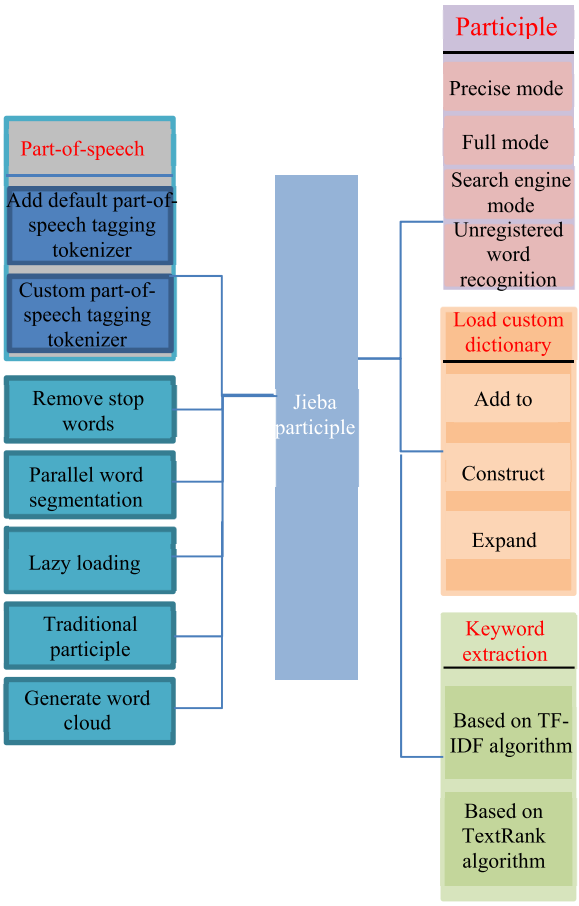

FIGURE 5. Functional structure diagram.

In the case of the graphical neural network, which calculates the strength of the relationship between companies, the data points correspond to a 0 by 1 matrix and thus satisfy the requirements for logical and operational operations. Where 0 indicates that each company keyword appears fewer than 10 times and 1 indicates that each company keyword appears 10 times or more. The calculated sum of each company node is therefore used to indicate the strength of the relationship between company nodes. Data with sums greater than 15 and sums greater than 20 and 25 are saved separately, as the relatively small amount of data with sums below 15 can have a significant impact on the experiment and the classification accuracy. Therefore, in order to perform the classification task more intuitively, the data with a sum less than 15 were discarded in order to obtain more convincing experimental results in order to complete the establishment of the edge relationship and conduct a comparison test to obtain the final experimental results.

IV. DATA PREPARATION

A. DATASETS

Suitable data is crucial for training the model and two datasets were chosen for this experiment, the Cora dataset [29] as well as the stock dataset.

The Cora dataset is a citation network comprised of papers on machine learning [30]. Each paper was initially divided into one of seven categories: case-based, genetic algorithms, neural networks, probabilistic methods, reinforcement learning, and rule-based learning.

The stock dataset was self-constructed using the acquired knowledge. The source of the data used to construct the dataset is the prospectus information of hundreds of stocks listed in the Science and Technology section of the Eastern Fortune website. Pharmaceutical manufacturing, computer technology, equipment manufacturing, materials manufacturing, metal manufacturing, electrical machinery, environmental protection, and rubber and plastics are the eight predetermined categories.

B. STOCK DATASET PRODUCTION

The dataset chosen for this occasion was created by learning itself, and the dataset consists of 273 company nodes and 1452 strong and weak relationships between company nodes. As preselected data objects, the data were extracted from pdf files of prospectuses of hundreds of listed companies on the Oriental Wealth website. The pdf files were converted to txt text files and input into the jieba word separation model for natural language processing. The number of keywords to mention is set by the model to 500, and keyword extraction is complete. This operation uses the TF-IDF algorithm in jieba splitting to calculate the probability of keyword occurrence and rank them in descending order. After the keywords have been extracted, they are imported into an Excel spreadsheet, less frequent and less influential keywords are deleted, and a 0,1 matrix is created for the input side based on the final confirmed keywords and the number of keyword occurrences. In addition, the types of companies corresponding to the 273 selected prospectuses must be determined in advance. They are artificially divided into eight categories: pharmaceutical manufacturing, computer technology, equipment manufacturing, material manufacturing, metal manufacturing, electrical machinery, environmental protection, and rubber-plastic manufacturing. After defining point relationships and category labels, edge connections are established. Logic and operations, which calculate the strength of the relationship between a company's nodes, are primarily responsible for establishing edge connections. The different companies' keywords are added with their corresponding 0's and 1's, and then the results for each keyword are added to produce a final result. The magnitude of the summation result, which allows for the establishment of side links, can be used to determine the strength of the relationship between the companies. Thus, both point and edge relations have been established, resulting in a complete graph structure that can be used as an experimentation input for the model. Figure 6 depicts a flowchart for illustratively depicting the exact steps. This creates the dataset for the classification of the stock. Table 1 displays the dataset's categories and their associated explanatory notes.

TABLE 1. Dataset affiliation and related explanatory notes.

Number of nodes	273
Number of edges	1452
Classifying categories	8
Classification	pharmaceutical manufacturing, computer technology, equipment manufacturing, material manufacturing, metal manufacturing, electrical machinery, environmental protection, rubber-plastic manufacturing.
Is there class imbalance	No

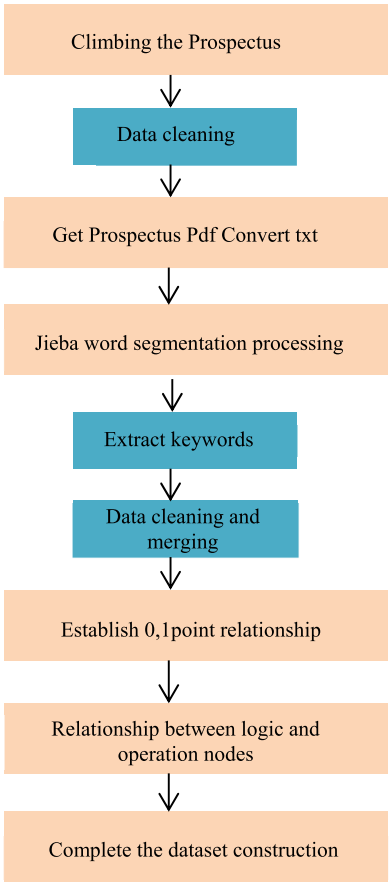

FIGURE 6. Diagram of data processing.

At the same time, in order to demonstrate the effect of logic and operations on this experiment more effectively, the data sides with summation results greater than 15 and greater than 20 and greater than 25 are also saved. The primary reason for discarding results with summation below 15 is that the

TABLE 2. Comparison of the accuracy of the training and validation sets.

Data summation	Training set loss	Training set accuracy	Validation set loss	Validation set accuracy
Greater than 15	0.4017	0.8667	0.7240	0.8333
Greater than 20	0.4528	0.8333	0.9500	0.7667
Greater than 25	0.3786	0.9000	1.0263	0.6600

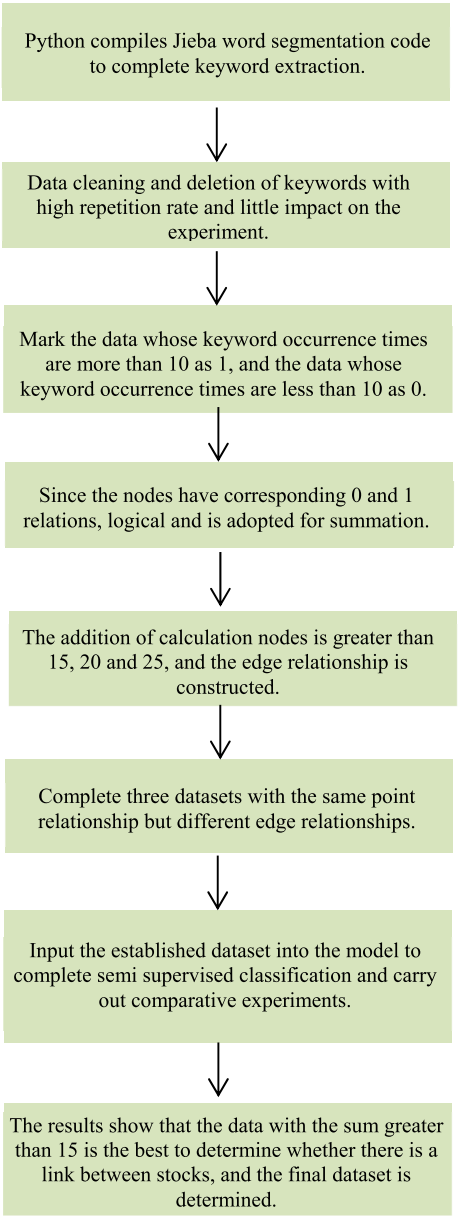

FIGURE 7. Method for generating datasets and experimental procedure.

experiment requires the summation of 368 primary keywords, and the relatively small amount of data with summation below 15 will have a significant impact on the experiment and the classification accuracy. In order to obtain more convincing experimental results, data with a sum of less than 15 were discarded in order to facilitate a more intuitive classification task. Consequently, a comparative experiment was conducted, and the final experimental results were derived and analyzed separately, as depicted in the analytical visualization in Figure 7.

C. DATASET PROCESSING

Initially, the data obtained by jieba splitting is preprocessed and tagged based on the frequency of occurrence. Include in an Excel spreadsheet and indicate as 1 data with more than 10 occurrences and as 0 data with fewer than 10 occurrences. Then, data with a high repetition rate and a low impact on the classification effect are eliminated and a standard point matrix is cleaned up. The primary objective of performing data pre-processing operations is to improve the model's recognition performance and learning efficiency, as well as the root classification accuracy. A further highlight of this experiment is that a minimal amount of data was selected for the training set, but accurate classification was still achieved. By dividing the dataset and adjusting the experimental procedure's parameters, the dataset was finally divided in a ratio of 1:1:8. 10% of the companies were selected at random for the training set, 10% for the validation set, and 80% for the test set. Additionally, the stocks were pre-classified into eight categories to assess the degree of accuracy of the final classification. Concurrently, to prevent a small amount of training data from being over-fitted and influencing the experimental results. In order to prevent overfitting, a Dropout layer is added to the final layer of the improved model this time. In order to determine if the model is overfitting, the dataset was split into the ratios 1:1:8, 5:1:4, 8:1:1, 1:3:6, 6:2:2, and 5:2:3 and compared using the model. This is to determine if the choice of a 1:1:8 ratio to divide the dataset for this experiment and obtain optimal results using a smaller dataset may have resulted in overfitting and had an effect on the experiment.

V. CONCLUSION AND ANALYSIS

A. DATASET SELECTION RESULTS AND ANALYSIS

Since the update of parameters such as features and weights of the input is unlikely to be the same for each training process in the PA-GCN model with the default activation function, the output of the program varies each time it is re-executed. Nevertheless, the differences between these results do not

TABLE 3. Comparison of accuracy and loss function of the test set.

Data summation	time/s	Loss functions	Accuracy
Greater than 15	4.0147	0.9428	80.75%
Greater than 20	4.7825	1.2789	76.53%
Greater than 25	4.0303	1.3300	69.95%

TABLE 4. Experimental results comparison of various activation functions in the model.

functions	Accuracy trends	Average accuracy of 10 runs
Elu	80%-82%	80.844%
Relu	78%-82%	79.575%
tanh	78%-80%	79.343%
sigmoid	44%-46%	45.070%

hinder the analysis and have no bearing on the experiment's conclusions. As a result, only the output of a single current program run is used for comparison experiments. The specific experimental comparison outcomes and accompanying analysis are displayed below.

The accuracy of the training set compared to the validation set as a result of the last iteration is shown in Table 2.

The accuracy and loss functions of the test set for the results of the last iteration are shown in Table3.

According to Table 2, accuracy of the training set through three experiments, selecting data with a summation greater than 15, with low loss and high accuracy in the validation set. When data greater than 25 are added and treated as edge relations, the accuracy of the training set is high, but the accuracy of the validation set is extremely low, indicating a clear case of overfitting. Therefore, when the selection sum is greater than 15, the relationship between the strength and weakness of the edges is more pronounced. The outcome will be superior to the outcomes of the other two selection scenarios.

Table 3 provides a comparison of the test set's accuracy and loss functions. It is possible to select data with a calculated sum greater than 15 based on the results of the test set, and with roughly equal time, the resulting loss function is the smallest, with a maximum accuracy of 80.75 percent. Therefore, the data selected for Table 3 with a sum greater than 15 are more accurate and precise.

Consequently, the experimental comparison of the various selection results demonstrates that the results obtained by selecting data with a calculated sum greater than 15 in the operation are more representative, and 80.75 % of the results demonstrate the experiment's success.

B. COMPARATIVE ANALYSIS OF MODIFIED ACTIVATION FUNCTIONS

The model's activation function will also have an effect on the experiment. In this instance, the default Relu activation function is replaced with the Elu activation function, which is an evolution of the Relu activation function, an activation function with negative values that can bring the average activation of the model close to zero, and the Elu activation function is not susceptible to gradient disappearance. Compared to the conventional Relu function, the Elu function utilized in this study can reduce training time and increase neural network precision proportionally. A significant contribution to the experimental findings. As the data crawled from each training set is random, the results of each classification will vary greatly during the training process. Therefore, in order to eliminate experimental chance and strengthen the experiment's credibility, the results of 10 consecutive calculations were chosen for each experiment and averaged. Table 4 displays the results obtained.

The results in Table 4 can be used to draw conclusions. When the model with the default activation function is modified for the experiments in this paper, the Elu activation function can improve the accuracy of the Relu activation function for 10 consecutive times by approximately 1.3% on average. A successful optimization and facilitation of the experimentally suggested PA-GCN model.

C. COMPARATIVE ANALYSIS OF OVER-FITTING PHENOMENA

To make this experiment more convincing and comprehensive. Verifying that the selected split dataset has not been overfitted and introducing a dropout layer on top of the PA-GCN model with a modified activation function yields the final PA-GCN model. This Dropout layer's function is to remove some neurons based on the corresponding probability, then train them, update the parameters and weights of the neurons that were not removed, and retain them. After updating the parameters, a portion of the neurons are removed and training is restarted. If the new neurons used for training have already been trained the first time, their parameters will continue to be updated, whereas the parameters of neurons that are removed for a second time, whose parameters have already been updated the first time, are not modified. This process is repeated until the nth batch is not removed when Dropout is performed. In addition, datasets with different division ratios of 1:1:8, 5:1:4, 8:1:1, 1:3:6, 6:2:2, and 5:2:3 were used for comparison experiments to evaluate the accuracy of the training and validation sets and the classification results obtained from experiments with different division ratios.

As the unified dataset for the experiments of this module, a stock dataset with a sum of 15 or more was used to establish edge relations. The accuracy of the training set with the results of the last iteration was compared with the validation set, and the final results were analyzed and conclusions were drawn. Table 5 displays the results of the experiments.

TABLE 5. Classification results for each dataset based on various division ratios.

Dataset splitting	Train set loss	Train set accuracy	Val set loss	Val set accuracy	Test Set Accuracy
1:1:8	0.4017	0.8667	0.7240	0.8333	81.69%
5:1:4	0.5892	0.8267	0.7370	0.8000	78.49%
8:1:1	0.6664	0.7936	0.6675	0.7778	89.29%
1:3:6	0.4325	0.8333	0.6829	0.8333	79.08%
6:2:2	0.5124	0.8078	0.6570	0.7667	84.85%
5:2:3	0.4687	0.8200	0.6444	0.8000	78.68%

Table 5 demonstrates that when the model is modified so that the proportion of the validation set remains unchanged, and the proportions of the training and test sets are adjusted to perform the experiments, the accuracy of the training set is approximately equivalent to that of the validation set, and the accuracy of the test set can be controlled to be greater than 78%. irrespective of the proportion of the subset used for the experiments. In addition, in order to better demonstrate that this experiment does not suffer from overfitting, the proportions of the training set were left unchanged and the proportions of the validation set were modified; the resulting data are presented in Table 5. The accuracy of the training and validation sets were roughly equivalent, and the accuracy of the final test set was greater than 78%. Additionally, it can be demonstrated that the small number of datasets used for training in this experiment to achieve the optimal classification accuracy are not overfit. It is desirable in both theory and practice and has been demonstrated.

D. COMPARATIVE MODEL ANALYSIS

This paper establishes two parts of model comparison experiments: supervised classification experiments with conventional machine learning on the Cora dataset and semi-supervised classification experiments based on the GCN model framework. The second phase consists of experimenting with the stock node dataset and incorporating the finalized dataset into the PA-GCN model proposed in this paper, followed by comparison tests with other traditional classification models and the standard GCN model.

A supervised experiment module within the Cora dataset, with 1900 entries serving as the training set and the remaining entries serving as the test set. In the semi-supervised experiment module of the GCN model, 200 training data, 400 validation data, and 1,000 test data are utilized. The last experiment's results were chosen for comparison and validation in each experiment, and the results are presented in Table 6.

In the stock node dataset for experiments, 30 bars were used as the training set, 30 as the validation set and 213 data as the test set. Semi-supervised classification of the nodes was completed using PA-GCN, where the model learning rate was set to 0.01, the weight decay was set to 5e-4 and the number of iterations was set to 200, and the results are shown in Table 7.

TABLE 6. Experimental outcomes for the dataset Cora.

Classification method	Accuracy（%）
SVM	71.0%
Logistic regression	71.7%
GraphSAGE	79.5%
GCN	80.1%
PA-GCN	81.2%

TABLE 7. Experimental results for classification of stock nodes.

Classification method	Accuracy（%）
SVM	75.86%
Logistic regression	76.20%
Random Forest	76.02%
GraphSAGE	78.40%
GCN	80.22%
PA-GCN	81.69%

Table 6 demonstrates that the GCN model outperforms conventional machine learning algorithms in certain classification problems, primarily because the model makes full use of the graph structure and the relationships between nodes and is able to better express the characteristics of nodes through multiple layers of learning. The classification

accuracy of 81.2% obtained here by PA-GCN in the Cora dataset also highlights that the model proposed in this paper does have some advantages.

The results of the comparison between the five model groups are presented in Table 7. The PA-GCN model proposed in this paper outperforms other models in terms of accuracy, with an accuracy rate of 81.69 percent. Traditional models rely heavily on manual feature engineering capture and have high experimental requirements. Consequently, the model proposed in this study can effectively increase the classification precision of stock classification.

VI. CONCLUSION

The graph convolutional neural network semi-supervised model (PA-GCN) proposed in this paper, which introduces the graph attention mechanism in advance and incorporates the Elu activation function, can compensate for the poor learning before the data enters the convolutional layer and can more effectively complete the classification by calculating the weights for the input data in advance. Incorporating graph neural networks into the financial industry is a daring experiment and a challenge. By creating the dataset on its own, the graph neural network is effectively integrated with natural language processing. The training was then completed utilizing the PA-GCN model, yielding varying experimental outcomes based on the edge relations and activation functions selected. Confirming this experiment demonstrates that selecting data with a summation greater than 15 is the optimal selection of edges, that the Elu activation function contributes well to the classification of the model, and that there is no overfitting. This paper concludes that the model proposed has a superior classification effect than other models. Lastly, the expected outcome of this paper is to improve the capabilities of economic systems through techniques related to graph neural networks, as well as to open up as much of an emerging field for graph neural networks as possible, so that graph neural networks can be combined with economic markets to take advantage of their unique benefits. Nonetheless, through this validation graph neural network has a distinct edge in classifying stocks from tiny samples. However, it is worthwhile to determine the efficacy of the classification strategy based on graph neural networks for big data sets. In my future academic career, I will also increase the stock data to determine how well the classification performs with greater amounts of data. This experiment will also pave the way for the application of graph neural networks in finance, and will hopefully provide scholars and researchers in related fields with useful ideas and methods. In the future, graph neural networks will be investigated further and more innovations will be developed.

REFERENCES

[1] Z. Huang, Y. Tang, and Y. Chen, "A graph neural network-based node classification model on class-imbalanced graph data," *Knowl.-Based Syst.*, vol. 244, May 2022, Art. no. 108538.

[2] J. Wang, "Text classification based on GNN," in *Proc. Int. Workshop Electron. Commun. Artif. Intell. (IWECAI)*, Jun. 2020, pp. 94–97, doi: 10.1109/IWECAI50956.2020.00026.

[3] B. M. Oloulade, J. Gao, J. Chen, T. Lyu, and R. Al-Sabri, "Graph neural architecture search: A survey," *Tsinghua Sci. Technol.*, vol. 27, no. 4, pp. 692–708, Aug. 2022, doi: 10.26599/TST.2021.9010057.

[4] J. Wu, J. He, and J. Xu, "DEMO-Net: Degree-specific graph neural networks for node and graph classification," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 406–415.

[5] B. Wang and J. Jia, "Semi-supervised node classification on graphs: Markov random fields vs. graph neural networks," in *Proc. AAAI*, 2021, pp. 10093–10101.

[6] B. Xu, J. Huang, L. Hou, H. Shen, J. Gao, and X. Cheng, "Label-consistency based graph neural networks for semi-supervised node classification," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2020, pp. 1897–1900.

[7] J. Yuan, H. Yu, M. Cao, M. Xu, J. Xie, and C. Wang, "Semi-supervised and self-supervised classification with multi-view graph neural networks," in *Proc. 30th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2021, pp. 2466–2476.

[8] X. J. Huang, Y. Y. Liu, and T. H. Ma, "A classification model for academic papers based on an improved graph neural network," *Data Anal. Knowl. Discovery.*, vol. 2022, pp. 1–14, Aug. 2022. [Online]. Available: http://kns.cnki.net/kcms/detail/10.1478.G2.20220413.1113.012.html

[9] C. Y. Qiang, X. G. Li, and X. Y. Ma, "Application of graph neural network in classification of bidding documents," in *Mini-Computer Systems*, 2022, pp. 1–7. [Online]. Available: http://kns.cnki.net/kcms/detail/21.1106.TP.20211112.1908.002.html

[10] S. Dabhi and M. Parmar, "NodeNet: A graph regularised neural network for node classification," 2020, *arXiv:2006.09022*.

[11] Y. Wu, Y. Song, and H. Huang, "Enhancing graph neural networks via auxiliary training for semi-supervised node classification," *Knowl.-Based Syst.*, vol. 220, Jan. 2021, Art. no. 106884.

[12] Y. Wang, Y. Shen, and D. Cremers, "Explicit pairwise factorized graph neural network for semi-supervised node classification," in *Uncertainty in Artificial Intelligence*. PMLR, 2021, pp. 1979–1987.

[13] B. Zhang, S. C. Song, and Y. H. Zhao, "GCN-based node classification study," *J. Hebei College Construct. Eng.*, vol. 2022, pp. 196–200, Jan. 2022.

[14] D. Y. Chen and J. L. Guo, "A graph attention-based method for classifying higher-order network nodes," *Comput. Appl. Res.*, vol. 2022, pp. 1–7, Nov. 2022, doi: 10.19734/j.issn.1001-3695.2022.09.0455.

[15] M. Gori, G. Monfardini, and F. Scarselli, "A new model for learning in graph domains," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Jul. 2005, pp. 729–734.

[16] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Dec. 2009.

[17] J. Bruna, W. Zaremba, and A. Szlam, "Spectral networks and locally connected networks on graphs," 2013, *arXiv:1312.6203*.

[18] P. W. Battaglia, J. B. Hamrick, and V. Bapst, "Relational inductive biases, deep learning, and graph networks," 2018, *arXiv:1806.01261*.

[19] G. Zhang, H. He, and D. Katabi, "Circuit-GNN: Graph neural networks for distributed circuit design," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 7364–7373.

[20] J. Zhou, G. Cui, and Z. Zhang, "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, Jan. 2018.

[21] M. He, Z. Wei, and J.-R. Wen, "Convolutional neural networks on graphs with Chebyshev approximation, revisited," 2022, *arXiv:2202.03580*.

[22] M. Liu, H. Gao, and S. Ji, "Towards deeper graph neural networks," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2020, pp. 338–348).

[23] Z. Ye, Y. J. Kumar, G. O. Sing, F. Song, and J. Wang, "A comprehensive survey of graph neural networks for knowledge graphs," *IEEE Access*, vol. 10, pp. 75729–75741, 2022, doi: 10.1109/ACCESS.2022.3191784.

[24] J. Z. Wang, L. W. Kong, and Z. C. Huang, "A review of graph neural networks," *Comput. Eng.*, vol. 47, no. 4, pp. 1–12, 2021, doi: 10.19678/j.issn.1000-3428.0058382.

[25] X. Q. Zeng, "Technology implementation of Chinese Jieba segmentation based on Python," *China Comput. Commun.*, vol. 18, pp. 38–39, Jan. 2019.

[26] X. Liu, Y. Su, and B. Xu, "The application of graph neural network in natural language processing and computer vision," in *Proc. 3rd Int. Conf. Mach. Learn., Big Data Business Intell. (MLBDBI)*, Dec. 2021, pp. 708–714, doi: 10.1109/MLBDBI54094.2021.00140.
[27] J. Sun, "Jieba Chinese text segmentation," Tech. Rep. +Version 0.42.1., 2020. [Online]. Available: https:// github.com/fxsjy/jieba
[28] J. Wang, X. Wang, C. Eckert, A. Subramaniyan, R. Das, D. Blaauw, and D. Sylvester, "A 28-nm compute SRAM with bit-serial logic/arithmetic operations for programmable in-memory vector computing," *IEEE J. Solid-State Circuits*, vol. 55, no. 1, pp. 76–86, Jan. 2020.
[29] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI Mag.*, vol. 29, no. 3, p. 93, 2008.
[30] L. Getoor, "Link-based classification," in *Advanced Methods for Knowledge Discovery From Complex Data*. London, U.K.: Springer, 2005, pp. 189–207.

YAOQUN XU received the B.S. degree in mathematics from Jilin University, Changchun, China, in 1993, the M.S. degree in mathematics from the Harbin Institute of Technology, Harbin, China, in 1997, and the Ph.D. degree in navigation, guidance, and control from Harbin Engineering University, Harbin, in 2002.

He is currently a Professor with the College of Computer and Information Engineering, Harbin University of Commerce. His current research interests include chaotic dynamics, neural networks, and intelligent optimization and decision.

YUHANG ZHANG received the bachelor's degree in engineering from the Harbin University of Commerce, Harbin, China, in 2021, where he is currently pursuing the master's degree with the School of Computer and Information Engineering, under the supervision of Prof. Yaoqun Xu. His current research interests include neural networks and graph neural network classification.