Journal of Network Intelligence ©2026 ISSN 2414-8105
Taiwan Ubiquitous Information Volume **11**, Number **1**, February **2026**

Deep Embedding across Node and Edge for Graph Anomaly Detection

De-Yang Zhang[1]

[1]The Scientific & Technological Information Center of Henan, Zhengzhou 450003, China
zhdy@qq.com

Ao-Jie Tan[2]

[2]School of Computer Science and Artificial Intelligence,
Zhengzhou University, Zhengzhou 450001, China
tanaojie@qq.com

Shu-Tang Liu[3,4]

[3]Faculty of Humanities and Arts,
Macau University of Science and Technology, Macau 999078, China
[4]Academic Affairs Office, Minjiang University, Fuzhou 350108, China
seahippo@126.com

Hao-Yi Fan[2]

[2]School of Computer Science and Artificial Intelligence,
Zhengzhou University, Zhengzhou 450001, China
fanhaoyi@zzu.edu.cn

Qiong Wu[5]

[5]Department of Art and Design,
Zhengzhou Professional Technical Institute of Electronic & Information, Zhengzhou 451450, China
295766890@qq.com

Fu-Quan Zhang[6,7,*]

[6]Fujian Provincial Key Laboratory of Information Processing and Intelligent Control,
Minjiang University, Fuzhou 350108, China
[7]Digital Media Art, Key Laboratory of Sichuan Province,
Sichuan Conservatory of Music, Chengdu 610021, China
8528750@qq.com

[*]Corresponding author: Fu-Quan Zhang

Received January 7, 2025, revised April 30, 2025, accepted July 20, 2025.

Abstract. *Currently, both the over-smoothing of feature embeddings and the interference of noisy edges during the feature aggregation process may make it more difficult for the model to distinguish between abnormal and normal nodes. Considering that the methods based on graph signal processing can effectively deal with the interference of noisy edges in the spectral domain, and the edge-embedding-based message passing networks in the spatial domain have improved the over-smoothing phenomenon to some extent. In this paper, by combining the two aspects mentioned above, we propose an effective GNN network. By combining the spectral graph neural network and the message passing network based on the spatial domain, the spectral graph neural network effectively solves the interference problem of noisy edges, and the edge-embedding-based message passing network in the spatial domain effectively solves the feature over-smoothing problem. Finally, we conduct training to increase the spatial distance between normal features and abnormal features. Extensive experiments have been carried out on multiple real-world graph datasets, Compared to the baseline models, our SPGNN model has achieved at least a* 5% *increase in AUROC on the YelpChi and T-Social datasets, and even more in some cases. On the YelpChi and T-Finance datasets, AUPRC has been improved by more than* 8%. *On the YelpChi, T-Finance, and T-Social datasets, MacroF1 has been enhanced by over* 2%.

Keywords: Graph neural network, Noisy edges, Spectral graph neural network, Message passing network, Spatial domain

1. Introduction.

Anomalies are typically defined as individual or group behavior patterns that deviate significantly from the majority of samples, manifesting as outliers or isolated clusters. In the graph data, anomalies can be considered as nodes, edges, or subgraphs that differ from the majority of objects within the graph. For example, in computer networks, anomalies may include newly introduced malicious users, suspicious access attempts, fraudulent reviews on shopping websites, or irregularities in financial transaction data. Given that anomalies can negatively impact the normal operation of various applications, Graph Anomaly Detection (GAD) has become an important area of research. In the real world, although there are many anomalous patterns, they are often mixed with normal data, making the anomalous patterns appear sparse. As a result, extreme class imbalance is a common phenomenon in Graph Anomaly Detection (GAD). Therefore, the well-known homophily assumption in Graph Anomaly Detection (GAD) suggests that normal nodes are more likely to interact with other normal nodes, while anomalous nodes tend to interact with other anomalous nodes. Based on the homophily assumption, many effective methods for anomaly detection have been developed, including unsupervised methods such as HomogCL[1] and NeCo[2], as well as supervised methods like FSGNN[3].

However, the homophily assumption may adversely affect anomaly pattern detection. Huang et al.[4] observed that removing heterophilic edges based on ground-truth labels resulted in monotonically decreasing model loss as heterophily levels diminished. The heterophilic edges defined as inter-class connections between nodes with divergent labels. In contrast, homophilic edges denote intra-class connections between nodes sharing identical labels, and a graph devoid of heterophilic edges is termed a homophily graph. Conventional graph neural networks (GNNs) enforce representation similarity among adjacent nodes, a mechanism that minimally impacts normal data patterns. However, for anomalous patterns, this homogenization distorts discriminative anomaly features, as excessive integration of normal node representations into anomalous nodes induces misclassification (e.g., anomalies labeled as normal)[5]. Crucially, heterophily constitutes a fundamental obstacle in graph anomaly detection (GAD). Huang et al.[4] empirically demonstrated that mere removal of all heterophilic edges improved the Simplified Graph Convolution

(SGC[5]) performance by approximately 30 percentage points over raw data baselines. Their methodology involved training an edge classifier to compute attribute variance between node pairs for edge typification, subsequently modifying graph topology and implementing node representation aggregation via GCN, achieving state-of-the-art efficacy. These findings confirm that heterophily impedes representation learning under classical message-passing paradigms. Within existing GAD frameworks, over-smoothing persists as a systemic challenge, predominantly stemming from over-reliance on homophily assumptions that coercively align node representations with their neighborhoods. This representation homogenization mechanism often yields suboptimal embeddings, particularly in real-world graphs containing both homophilic and heterophilic interactions. For instance, in social networks, a user may share congruent interests with certain peers while exhibiting divergent preferences with others. Furthermore, real-world graphs inherently contain stochastic uncertainties and noisy links induced by adversarial attacks or systemic noise. Indiscriminate smoothing across all edges risks eroding node distinguishability, thereby degrading the generalizability and robustness of graph representation learning. Consequently, a critical research frontier lies in developing heterophilic edge mitigation strategies to address over-smoothing. Prevailing methodologies encompass graph attention mechanisms[6, 7], task-specific optimization objectives[8], and edge prediction architectures[8, 9]. These approaches aim to circumvent limitations of conventional GNNs by enhancing identification and utilization of critical structural features, ultimately advancing GAD performance.

Existing methods can be roughly divided into two categories. One is the method based on graph signal processing, and the other is the method that conducts message passing through spatial structure selection strategies. The core idea of the methods based on graph signal processing is that the Laplacian matrix reflects the structural information of the graph. Through operations such as eigen-decomposition on the Laplacian matrix, the spectral information of the graph can be obtained. This spectral information contains important features and structural attributes of the graph and can be used as an important basis for anomaly detection or classification. Although the networks based on spectral analysis theory have seen significant improvements in credibility and efficiency to some extent, in general, industrial graph data in the real world is usually quite large. For large graphs or those with complex structures, the computational complexity of calculating the eigenvectors of the graph's Laplacian matrix is extremely high, which will consume a large amount of computing resources and time. Meanwhile, the performance of spectral graph neural networks highly depends on the graph's Laplacian matrix, and the Laplacian matrix, in turn, depends on the structure of the graph. If the structure of the graph changes, such as the addition or deletion of nodes or the alteration of edge weights, the Laplacian matrix will also change accordingly, which may lead to a decline in the model's performance. The core idea of the methods that conduct message passing through spatial structure selection strategies is to carry out message passing by designing neighbor aggregation functions, and by learning and updating the implicit embedding vectors of nodes, a good model for downstream tasks can be achieved. Although message passing networks based on the spatial domain have, to some extent, made the understanding much easier compared to spectral methods and are more flexible and can also achieve good results at the same time, they ignore the effective information in the spectral domain. Moreover, the simple superposition processing of neighbor information through various algorithms will also lead to the over-smoothing phenomenon of the finally obtained node features, which can easily increase the uncertainty of the results of anomaly detection.

To address the aforementioned issues, this paper proposes a combined approach that integrates graph signal processing techniques with spatial structure-based message passing methods. Firstly, since removing heterophilic edges requires a large number of ground truth labels, which are often unavailable in real-world scenarios, we opt to approximate these ground truth labels using the predictions from spectral graph neural networks. Subsequently, we use these approximated ground truth labels to perform a series of computations to remove a certain proportion of heterophilic edges, thereby enhancing the data. Secondly, the choice of spectral graph neural networks for the first step is due to their ability to avoid the accumulation of prediction errors. If spatial domain-based message-passing networks were used in the first step, the network's own predictions would typically be considered as part of the second step's results. This is akin to having a spatial domain-based message-passing network acting as both the referee and the player. Thirdly, to address the issue of over-smoothing, the proposed method of extracting node features separately based on edge relationship types and then concatenating them is effective. Finally, since the network simulates real-world data, for limited-supervision networks, this paper introduces the use of the sphere loss in the training of the second step's spatial domain-based network. This approach significantly enhances the network's performance.

The main contributions of this paper are as follows:

(1) To address the common issues of over-smoothing and noisy edges in feature aggregation, this paper proposes an optimized algorithm called SPGNN that combines information from the spectral domain and the spatial domain. The algorithm is mainly composed of a method for dealing with noisy edges in the spectral domain and an algorithm for handling over-smoothing in the spatial domain.

(2) Based on the idea of deep one-class anomaly detection, a spherical loss function is proposed. This loss function mainly uses the mean of batch samples to calculate the centroid, and during training, it pulls normal samples closer to the centroid while pushing abnormal samples farther away from the centroid.

1.1. **Realted work.**

1.1.1. *Spectral Graph Neural Network.* In recent years, spectral networks have gained significant credibility in interpretability due to the support of graph signal processing theory. The core idea is that the Laplacian matrix reflects the structural information of the graph. Among them, SGCN[10] is based on spectral graph theory and utilizes the spectral information obtained from the Laplacian matrix of the graph and its eigendecomposition to design the convolution operation. This unique design endows SGCN with strong expressiveness and adaptability when processing graph data. He et al.[11] achieved the convolution operation on the graph by performing the Chebyshev polynomial expansion of the Laplacian matrix of the graph, which can efficiently capture the local and global structural features of the graph as well as the complex relationships between nodes. The most famous one is GCN. It draws on the ideas of the traditional convolutional neural network (CNN) and can be combined with many methods proposed in other fields, with generally good effects, such as the attention mechanism and recurrent neural networks. However, in these experiments, the existence of anomalies has not been observed to lead to the "right shift" phenomenon, that is, the concentration of spectral energy distribution in the low frequency is reduced, while the concentration in the high frequency is increased. Therefore, Tang et al.[12] proposed the Beta Wavelet Graph Neural Network (BWGNN). BWGNN has band-pass filters with spectral and spatial localization, which can better handle the "right shift" phenomenon in anomalies and better capture the energy in the high frequency part. Lu et al.[13] observed that the normalized Laplacian matrix often has repeated eigenvalues. To reduce the repetition of eigenvalues, they proposed an eigenvalue correction strategy. This strategy involves sampling new eigenvalues u at equal intervals from the interval $[0, 2]$ and combining them with the original eigenvalues λ to generate new eigenvalues μ. However, these methods have not fully utilized the information in relationships on multi-relational datasets.

1.1.2. *Message Passing Networks Based on the Spatial Domain.* Message Passing Neural Networks (MPNNs) remain one of the core methods for handling graph-structured data, and numerous innovations and improvements have been made on this basis. For example, GraphSAGE [14] introduces a neighborhood aggregation method, this method controls the size of the subgraph through neighborhood uniform sampling, applies mini-batch-based deep model training to the training of large-scale graphs, which makes it possible to train extremely large-scale graphs. In contrast, GCN uses a full-graph training method. GAT[15] theoretically breaks through the traditional way of explicitly defining neighbors through the adjacency matrix, instead using an attention distribution mechanism to measure the distance between nodes. SGFormer[16] captures full node-pair interactions through a single-layer global attention mechanism while maintaining linear complexity, avoiding the high computational complexity of traditional multi-layer attention mechanisms. Yang et al.[17] by introducing additional Fairness Facilitating Features (F3), the sensitive biases within node representations can be neutralized. This approach aims to improve the model's fairness metrics while maintaining a high level of predictive accuracy. However, none of their works have considered the information in edge relationships, and noisy edges also exacerbate the difficulty of node anomaly detection.

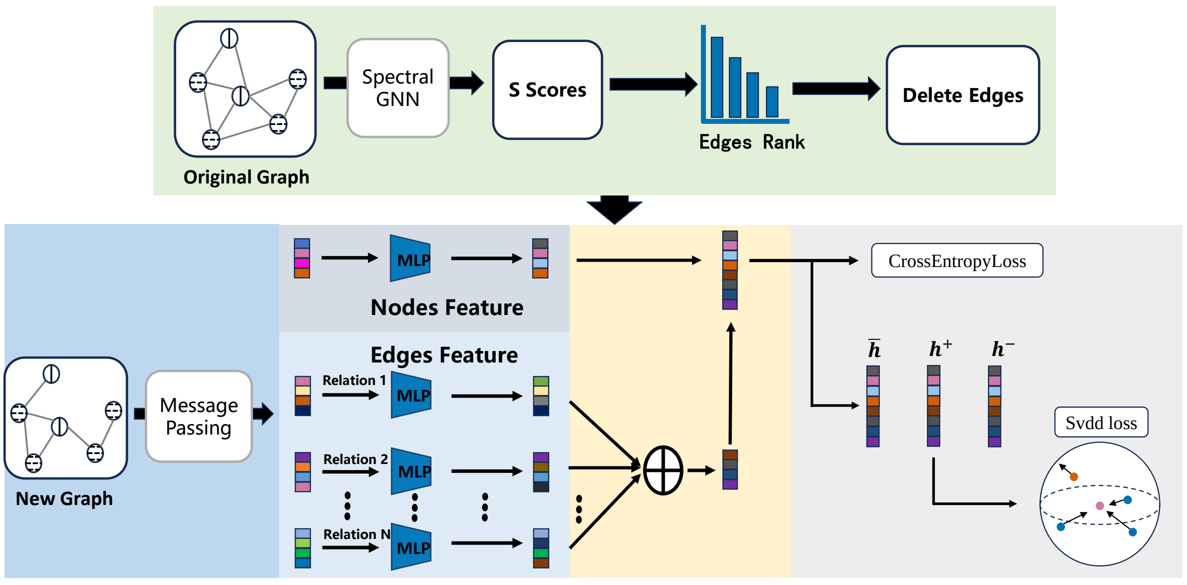

Figure 1. The proposed framework of SPGNN

2. **Method.** In this section, we provides a detailed description of the overall framework for anomaly detection on graph nodes. The framework achieves effective low-dimensional feature representations by removing heterophilic edges and leveraging edge relationship types.

2.1. **Overall Framework.** The framework of the SPGNN (Spectral Graph Neural Network) proposed for graph anomaly detection is illustrated in Figure 1. The model is divided into two parts. The first part focuses on denoising the original graph, which involves systematically removing heterophilic edges to obtain a new graph. A spectral graph neural network is used without any training to provide unbiased scoring for the nodes in the original graph. Each node receives a confidence score, which indicates the likelihood of the node being either a normal node or an anomaly. Using the confidence scores, we perform an inner product calculation with the aggregated scores S mentioned earlier. The resulting scores are then ordered, with the lowest scores identifying the heterophilic edges. By removing a certain proportion of these heterophilic edges, we obtain a purified graph with reduced edge noise. (Here, "purified" refers to the graph after ordered removal of noisy edges, relative to the original graph, as heterophilic edges are detrimental to the representation of homophilic graph features.) The reason for removing only a certain proportion of edges is that the confidence scores are derived directly from the model without any training and thus cannot be fully trusted. However, they can still serve as a part of the optimization process. The second part involves using a spatial-domain message-passing network to learn from the purified graph obtained. Since there are multiple types of edge relations in the datasets, in order to make full use of the information on the edge relations, it is considered to embed different types of edges separately and then fuse them. Finally, they are spliced with the attribute features of the target node to form a complete feature embedding of the target node. After obtaining the complete embedding of the target node, we perform a mean-processing on all node embeddings in each round to obtain a central node, that is, the center of the spherical loss. At the same time, in the supervised training process, normal nodes and abnormal nodes will be taken out and processed with the center by svdd loss. This loss will bring the embeddings of normal nodes closer and at the same time expand the distance between abnormal nodes and the center. The entire model obtains the optimal node representation by enabling the target nodes to obtain node embeddings with more sufficient information, and by using the loss to shorten the distance between normal nodes and increase the distance between abnormal nodes for node-level anomaly detection finally.

2.2. **Edge Noise Reduction.** A graph can be represented as $G = \{\boldsymbol{V}, \boldsymbol{E}, \boldsymbol{X}\}$, where $\boldsymbol{V}$ is the set of nodes, $\boldsymbol{E}$ is the set of edges, and $\boldsymbol{X} \in \mathbb{R}^{|V| \times d_X}$ is the feature matrix of nodes.

In homogeneous graphs, high-pass filters are very important for detecting heterogeneous edges. Generally speaking, suppose $\boldsymbol{A}$ is the adjacency matrix. Then the graph Laplacian $\boldsymbol{L}$ can be expressed as $\boldsymbol{D} - \boldsymbol{A}$ or $\boldsymbol{I} - \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}}$. The eigenvalue λ_i of $\boldsymbol{L}$ generally represents the high frequency of the graph. The k_{th} power of the normalized graph Laplacian operator is a commonly used high-pass filter. Here, the heterogeneity of nodes is defined within the 1-hop neighborhood. Then we adopt the aggregation of the label distribution of the 1-hop neighborhood:

$$S = \widehat{\boldsymbol{L}}\boldsymbol{Y} \tag{1}$$

Among them, the i_{th} row of S is called the aggregated similarity score of node i. $\widehat{\boldsymbol{L}}$ is the random walk normalized graph Laplacian (with self-loops added), and $\widehat{\boldsymbol{L}} = \boldsymbol{I} - \widetilde{\boldsymbol{D}}^{-1}\widetilde{\boldsymbol{A}}$ can be written as:

$$\begin{pmatrix} \frac{d_1}{d_1+1} & -\frac{1}{d_1+1} & \cdots & -\frac{1}{d_1+1} \\ -\frac{1}{d_2+1} & \frac{d_2}{d_2+1} & \cdots & -\frac{1}{d_2+1} \\ \vdots & \vdots & \ddots & \vdots \\ -\frac{1}{d_N+1} & -\frac{1}{d_N+1} & \cdots & \frac{d_N}{d_N+1} \end{pmatrix} \tag{2}$$

The diagonal element $\widehat{L}_{ii} = \frac{d_i}{d_i+1}$. If there is an edge between node A and node B, then the off-diagonal element $\widehat{L}_{ij} = -\frac{1}{d_i+1}$. If there is no edge between node A and node B, then $\widehat{L}_{ij} = 0$.

When performing edge-level scoring on the original graph, the aggregated scoring function can be written in the following form:

$$hetero(i) = \frac{1}{|\mathcal{N}(i)|}|\{u \colon u \in \mathcal{N}(i), y_u \neq y_i\}| \tag{3}$$

$$S_i = SIGN * [\frac{d_i}{d_i+1}hetero(i), -\frac{1}{d_i+1}hetero(i)] \tag{4}$$

Among them, the $SIGN$ of a normal node is represented as 1, the $SIGN$ of an abnormal node is represented as -1, and $hetero(i)$ represents the heterogeneity rate of node i. There are two confidence levels for node prediction (the confidence level biased towards normal nodes and the confidence level biased towards abnormal nodes). This is because abnormal nodes have high heterogeneity while normal nodes have low heterogeneity. So the aggregated scores can be used for the inner product as follows, and the edge with the smallest score is the heterogeneous edge (connecting nodes of different types):

$$S_{v \in \boldsymbol{V}_a} \cdot S_{v \in \boldsymbol{V}_a} > S_{v \in \boldsymbol{V}_n} \cdot S_{v \in \boldsymbol{V}_n} > 0 > S_{v \in \boldsymbol{V}_a} \cdot S_{v \in \boldsymbol{V}_n} \tag{5}$$

among them, V_a and V_n represent abnormal nodes and normal nodes respectively.

Finally, we obtain the prediction $\hat{\boldsymbol{Y}}$ from the input original graph. According to the high-frequency indicators of this prediction, the edge score $\boldsymbol{E}$ is given. Finally, the edge scores are sorted to delete heterogeneous edges in an orderly manner. Here, $\boldsymbol{L}$ is the graph Laplacian matrix, and $\hat{\boldsymbol{Y}}$ is the obtained prediction result. The edge scores are calculated by transposing both $\boldsymbol{L}$ and $\hat{\boldsymbol{Y}}$, and then performing matrix multiplication. The edge score $\boldsymbol{E}$ is expressed as follows:

$$\boldsymbol{E} = \boldsymbol{L}\widehat{\boldsymbol{Y}}\widehat{\boldsymbol{Y}}^T\boldsymbol{L}^T \tag{6}$$

2.3. **Edge Relationship Embedding.** When performing feature embedding on the new graph after data augmentation, we consider that most of the current methods obtain the feature embedding of the node itself by performing a series of processing on the data of the surrounding nodes of the node. However, in the datasets of this article, there is actually more information that has not been effectively utilized, such as the relationship types of edges. Accordingly, after performing message passing processing on the new graph, each node and edge has data features. We traversed the edge relationship types respectively using a multi-layer perceptron (Before this, many papers' experiments have proven that the multi-layer perceptron has relatively excellent processing capabilities for graph data). Finally, all the information is spliced together. The formula is shown as follows:

$$\boldsymbol{h}_n, \boldsymbol{h}_e = MessagePassing(\cdot) \tag{7}$$

$$\boldsymbol{h}'_n, \boldsymbol{h}'_e = \mathrm{MLP}(\boldsymbol{h}_n, \boldsymbol{h}_e) \tag{8}$$

$$\boldsymbol{h} = \mathrm{CAT}(\boldsymbol{h}'_n, \boldsymbol{h}'_e) \tag{9}$$

where $\boldsymbol{h}_n$ and $\boldsymbol{h}_e$ represent the information of nodes and edges after message passing respectively. $\boldsymbol{h}'_n$ and $\boldsymbol{h}'_e$ represent the information of nodes and edges after being processed by the multilayer perceptron. The CAT function is defined as performing vector addition between paired vectors, where we have executed vector-wise summation operations. The feature embedding of the complete node obtained finally is represented by $\boldsymbol{h}$.

Algorithm 1 SPGNN

```
Input: Nodes $\boldsymbol{V}$, adjacency matrix $\boldsymbol{A}_{adj}$, original graph data $G$, spectral graph filter $G_{sg}$, train epoch $T$.
 1: Input the data into the untrained spectral graph filter to obtain the predicted value $\widehat{Y} = G_{sg}(G, A_{adj})$.
 2: Calculate the Laplacian matrix of the graph $\boldsymbol{L} = \boldsymbol{I} - \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}}$.
 3: Calculate the edge scores based on the predicted values $\boldsymbol{E} = \boldsymbol{L}\widehat{\boldsymbol{Y}}\widehat{\boldsymbol{Y}}^T\boldsymbol{L}^T$.
 4: Set the deletion ratio $r$, delete according to the ranking of $\boldsymbol{E}$, and obtain the new graph $G'$.
 5: while $T < 101$ do
 6:     Through message passing obtain node original features $\boldsymbol{h}_n$ and edge original features $\boldsymbol{h}_e$
 7:     Further train and extract $\boldsymbol{h}_n$ and $\boldsymbol{h}_e$ through the MLP network obtain $\boldsymbol{h}'_n$ and $\boldsymbol{h}'_e$.
 8:     $h \leftarrow \mathrm{CAT}(\boldsymbol{h}'_n, \boldsymbol{h}'_e)$ Perform vector addition on edge and node features.
 9:     Then directly use the Softmax function for scoring.
10: end while
Output: Scores for normal and abnormal nodes $y$.
```

2.4. **Loss function and scoring function.** When choosing the loss function, considering that the embedded feature $\boldsymbol{h}$ in the homogeneous graph is relatively long even in a low-dimensional space, the idea of hypersphere in Deep SVDD is thought of. In simple terms, that is, let the normal data be close to the center of the sphere and let the abnormal data be far away from the center of the sphere. $\boldsymbol{h}_c$ is mainly obtained by obtaining the center of an n-dimensional feature, and then obtaining the mean value of the n-dimensional feature of normal data and the mean value of the n-dimensional feature of abnormal data in each round respectively, and performing the following calculations:

$$\boldsymbol{h}_c = \overline{\boldsymbol{h}} \tag{10}$$

$$\mathcal{L}_{svdd} = \mathrm{dist}(\boldsymbol{h}^+ - \boldsymbol{h}_c) + \mathrm{dist}(\boldsymbol{h}^- - \boldsymbol{h}_c) \tag{11}$$

where $\overline{\boldsymbol{h}}$ represents the mean of all embedded features, $\boldsymbol{h}^+$ represents the mean of normal data of embedded features, and $\boldsymbol{h}^-$ represents the mean of abnormal data of embedded features.

Let $x_v \in \mathbb{R}^{d_X}$ represent the feature vector of node v. Given a graph encoder $g(\cdot;\beta_g)$ parameterized by β_g, embed each node v into a low-dimensional representation $h_v \in \mathbb{R}^d$, such as $h_v = g(v;\beta_g)$. GAD can be conceptualized as a binary classification task where nodes are divided into two categories, including normal (majority) and abnormal (minority) classes. Generally speaking, given the representation of node v, denoted as $\boldsymbol{h}_v$, a predictor $P(\cdot;\beta_g)$ can be used for prediction, denoted as $\boldsymbol{p}_v \in \mathbb{R}^K$, as follows:

$$\boldsymbol{Scores} = \boldsymbol{p}_v = P(h_v;\beta_g) = \mathrm{SOFTMAX}(\boldsymbol{W}_p\boldsymbol{h}_v + \boldsymbol{b}_p) \tag{12}$$

where β_g are learnable parameters. In the case of anomaly detection here, $K = 2$ indicates the number of node categories. For a labeled node v, let $\boldsymbol{y}_v \in \mathbb{R}^K$ denote the label vector, where $\boldsymbol{y}_v[k] = 1$ if and only if node v belongs to class k. Typically, given the training set V_{train}, the loss function is obtained by taking the cross-entropy loss, as shown below:

$$\mathcal{L}_{ce} = -\sum_{v\in V_{tr}}\sum_{k=0}^{K-1} y_v[k]\ln p_v[k] \tag{13}$$

In summary, the pseudo-code of SPGNN is shown in **Algorithm** 1, our loss function is expressed as follows:

$$\mathcal{L} = \mathcal{L}_{svdd} + \mathcal{L}_{ce} \tag{14}$$

3. **Experiment.** In this section, we first introduce the datasets, evaluation metrics, baseline methods, and implementation details. Subsequently, we compare the experimental results of SPGNN and the baseline methods on four datasets to verify the effectiveness of SPGNN.

3.1. **Datasets.**

Table 1. Statistics of the used datasets.

	Nodes	Edges	Features	Anomaly	Train:Valid:Test
Amazon[18]	11,944	4,398,392	25	6.87%	1%:33%:66%
YelpChi[19]	45,954	3,846,979	32	14.53%	1%:33%:66%
T-Finance[12]	39,357	21,222,543	10	4.58%	1%:33%:66%
T-Social[12]	5,781,065	73,105,508	10	3.01%	1%:33%:66%

We conducted experiments on the four datasets introduced in Table 1:

- **Amazon(McAuley and Leskovec, 2013)** [18] aims to find abnormal users who write false product reviews for payment by merchants under products in the musical instrument category on the website Amazon.com. It also has three relationships: U-P-U (users review at least one same product), U-S-U (users have at least one same star rating within a week), and U-V-U (users with the highest 5% mutual review similarity).
- **YelpChi(Rayana and Akoglu, 2015)** [19] aims to find abnormal reviews on the website Yelp.com that unjustly promote or downgrade certain products or businesses. There are a total of three edge types in the original graph, including R-U-R (reviews posted by the same user), R-S-R (reviews under the same star rating and the same product), and R-T-R (reviews posted under the same product in the same month).
- **T-Finance(Tang et al., 2022)** [12] aims to find abnormal accounts in the transaction network. The nodes in the original graph are unique anonymous accounts with 10-dimensional features related to the number of days of registration, log activities, and interaction frequency. The edges in the graph represent two accounts with transaction records. If a node belongs to categories such as fraud, money laundering, and online gambling, human experts will annotate the node as abnormal.
- **T-Social(Tang et al., 2022)** [12] aims to find abnormal accounts in social networks. It has the same node annotations and features as T-Finance, and two nodes are connected if they maintain a friendship relationship for more than three months. The scale of T-Social is 100 times larger than that of YelpChi and Amazon.

3.2. **Baselines.**

We selected ten state-of-the-art baselines to compare with our method, thereby evaluating its effectiveness comprehensively. Details of these baselines are as follows:

- **MLP (multi-layer perceptron (Rosenblatt, 1958))** [20]**:** A multi-layer perceptron network with one hidden layer and ReLU activation.
- **GCN (Graph Convolutional Network (Kipf and Welling, 2017))** [21]**:** A graph neural network that performs graph spectral convolution through the local first-order approximation of the spectral filter.

- **GraphSAGE (Graph Sample and AggregatE (Hamilton et al., 2017))** [14]**:** A graph neural network that samples and aggregates neighboring features to generate node embeddings. It also proposes three aggregation methods: mean, LSTM, and pooling. In the experiment, the mean aggregator is used.
- **GAT (Graph Attention Networks (Veličković et al., 2018))** [15]**:** A graph neural network that applies an attention mechanism to the neighborhood aggregation process.
- **GraphConsis ((Liu et al., 2020))** [22]**:** This method identifies three inconsistency problems in the graph anomaly detection task, namely context inconsistency, feature inconsistency, and relationship inconsistency. To deal with the context inconsistency problem, this work assigns a learnable context embedding to each node to capture its local structure. To deal with feature inconsistency, this work filters neighbors according to the consistency score estimated by the neural classifier. To solve the relationship inconsistency problem, this work trains an embedding for each relationship and uses a self-attention mechanism to aggregate neighbors.
- **PC-GNN (Pick and Choose Graph Neural Network (Liu et al., 2021b))** [23]**:** This method uses a label-balanced sampler to select nodes and edges for training, where the sampling probability is inversely proportional to the label frequency. In addition, it also proposes a neighborhood sampler that oversamples the neighborhood of fraudulent nodes and undersamples the neighborhood of normal nodes.
- **BWGNN (Beta Wavelet Graph Neural Network (Tang et al., 2022))** [12]**:** This method discovers that anomalies can cause the energy in the graph domain to transfer from the low-frequency part to the high-frequency part. It uses the Beta kernel to create band-pass filters with spatial and spectral locality for anomaly detection. For multi-relational graphs, BWGNN provides two options, namely homogeneous and heterogeneous. BWGNN (homo) transforms a multi-relational graph into a single graph for convolution, while BWGNN (hetero) applies convolution to each relationship separately and aggregates the output of each relationship through maximum pooling.
- **H2-FDetector (Graph Neural Network-based Fraud Detector with Homophilic and Het-erophilic Interactions (Shi et al., 2022))** [24]**:** This method detects homo-edges and hetero-edges through an auxiliary neural classifier, which is trained with an additional loss function. It adopts different aggregation strategies for the detected hetero-edges, in which the opposite values of the node embeddings are used. Moreover, it averages the embeddings within each class and uses the average embedding as the class prototype. The node representation needs to be close to its corresponding class prototype.
- **GDN (Graph Decomposition Network (Gao et al., 2023b))** [25]**:** This method divides node representations into dissatisfaction with class features and surrounding features. For class features, it applies class constraints to ensure that nodes within the same class have similar class features. For surrounding features, it applies connectivity constraints to ensure that adjacent nodes have similar surrounding features.
- **GAGA (Group AGgregation enhanced TrAnsformer (Wang et al., 2023))** [26]**:** This method explicitly uses partially observed labels to divide neighbors into three groups: normal, fraudulent, and unknown nodes. Each group of nodes is processed differently during processing. It enhances node features through trainable skip, relationship, and group embeddings, and uses a Transformer encoder to transform node features.

Table 2. Anomaly detection performance of all methods. The best results are marked in **bold**, and all values are percentages (%).

Dataset	**Amazon**			**YelpChi**		
	AUROC	**AUPRC**	**MacroF1**	**AUROC**	**AUPRC**	**MacroF1**
MLP[20]	92.39	79.37	87.53	72.18	31.09	61.61
GCN[21]	87.34	48.06	70.94	54.65	17.07	35.59
GraphSAGE[14]	90.12	73.17	84.25	73.7	34.57	63.33
GAT[15]	80.74	45.46	63.45	70.14	28.9	61.22
GraphConsis[22]	64.23	21.38	55.35	78.91	38.4	64.96
PC-GNN[23]	91.18	77.92	85.25	75.17	36.6	64.23
BWGNN(homo)[12]	88.56	79.26	**90.48**	72.15	30.93	61.24
BWGNN(hete)[12]	84.64	64.0	80.41	77.62	39.87	66.54
H2-FDetector[24]	83.81	49.37	72.46	72.38	32.37	63.97
GDN[25]	92.16	**81.87**	89.75	75.92	38.04	64.81
GAGA[26]	82.61	56.59	76.85	71.61	31.96	61.81
Ours	**93.59**	80.42	86.75	**83.92**	**48.12**	**70.25**

Table 3. Anomaly detection performance of all methods. The best results are marked in **bold**, and all values are percentages (%). (OOM represents out of memory)

Dataset	**T-Finance**			**T-Social**		
	AUROC	**AUPRC**	**MacroF1**	**AUROC**	**AUPRC**	**MacroF1**
MLP[20]	92.17	52.79	82.33	66.95	6.0	54.09
GCN[21]	89.29	53.94	77.16	83.3	23.79	65.16
GraphSAGE[14]	89.42	49.08	77.62	71.45	8.73	56.47
GAT[15]	87.4	45.6	75.49	73.46	13.47	61.98
GraphConsis[22]	92.61	70.7	85.37	OOM	OOM	OOM
PC-GNN[23]	91.74	74.77	86.97	64.68	4.3	49.66
BWGNN[12]	93.08	77.79	86.97	84.4	49.96	76.37
H2-FDetector[24]	OOM	OOM	OOM	OOM	OOM	OOM
GDN[25]	88.75	54.27	76.62	67.69	7.51	55.76
GAGA[26]	92.36	64.34	81.1	78.92	23.72	65.58
Ours	**95.81**	**86.13**	**89.58**	**94.86**	**51.13**	**78.32**

3.3. **Experimental design and metrics.**

3.3.1. *Experimental parameters.* The key hardware parameters of the experimental platform in this paper are as follows: CPU-Intel(R) Xeon(R) Gold 5220R, 2.20 GHz, 256 GB RAM; GPU-NVIDIA 3090, 24 GB RAM. The experimental environments include PyTorch and Deep Graph Library. According to the size of the datasets, we set the training proportion of Amazon, YelpChi, and T-Finance to 1%, and the training proportion of T-Social to 0.01%. In all scenarios, the remaining data is divided into validation and test according to the ratio of 1 : 2. The training epochs for the four datasets are all set to 100, the learning rate is set to 0.001, and the batch-size is set to 128. Although it is supervised training, the labeled data used here is only one percent of the total data. Meanwhile, in the optimal results, the deletion ratios of heterogeneous edges in Amazon, YelpChi, T-Finance, and T-Social are set as 0.0174, 0.2, 0.0138, and 0.0162 respectively. The detailed dataset proportion setting is shown in Table 1.

3.3.2. *Metrics.* We use the AUROC score (the Area Under a Receiver Operating Characteristic Curve), the AUPRC score (the Area Under the Precision Recall Curve) and the MacroF1 score (Macro averaged F1) to evaluate the anomaly detection performance of different models.

3.4. **Experimental results.**

3.4.1. *Performance analysis.* In this section, we quantitatively evaluate the anomaly detection performance of our proposed method SPGNN. The results of all the compared methods on each dataset are presented in Table 2 and Table 3.

We illustrate the effectiveness of the proposed method by presenting the results of the model on the graph data anomaly detection task. On the YelpChi dataset, compared with other anomaly detection methods, the AUROC, AUPRC and MacroF1 metrics of our method are all the highest, exceeding the best comparison method by 5.01%, 8.25% and 3.71% respectively. On the T-Finance dataset, they exceed the best comparison method by 2.73%, 8.34% and 2.61% respectively. On the T-Social dataset, they exceed the best comparison method by 10.46%, 1.17% and 1.95% respectively. The performance on the Amazon dataset is not satisfactory, possibly due to the large number of edges and relatively fewer nodes in the dataset. This extreme imbalance may lead to most nodes having similar representations, causing almost all neighboring nodes to exhibit alike expressions regardless of whether they are anomalous or normal. This phenomenon increases the difficulty of node anomaly detection. In summary, the method proposed in this paper can effectively extract features from graph data and demonstrates the effectiveness of this approach in anomaly detection.

3.4.2. *Ablation study.* To verify the effectiveness of each part of the model, ablation experiments are carried out on the Amazon dataset, YelpChi dataset, T-Finance dataset, and T-Social dataset respectively in this paper. The two model structures are:

- **wo-***aug*: Compared with the entire SPGNN, the edge deletion module for data enhancement on the original graph is not added.
- **wo-***svdd*: Compared with the entire SPGNN, the spherical loss module for processing positive and negative sample embeddings is not added.

We conducted experiments on three models: the model with the GNN backbone network and the spherical loss module, the model with the GNN backbone network and the homogeneous graph augmentation module (DA), and the model with the GNN backbone network, the homogeneous graph augmentation module (DA), and the spherical loss module. It can be seen that on the T-Finance and T-Social datasets, the model with only the GNN backbone network and the spherical loss module performs almost as well as the model with the GNN backbone network, the homogeneous graph augmentation module (DA), and the spherical loss module. However, on the Amazon and YelpChi datasets, the performance is not good. We analyzed that this might be because T-Social has millions of nodes, while the Amazon and YelpChi datasets only have tens of thousands of nodes. When selecting one percent of the data for training, T-Social has enough nodes to calculate an approximate centroid, allowing normal and abnormal nodes to deviate from and approach each other well. In contrast, the Amazon and YelpChi datasets only have a few hundred nodes to calculate the centroid (i.e., the mean of features), which is not sufficient to simulate the correct centroid. However, this still proves the effectiveness of the spherical loss module. Meanwhile, the model with the GNN backbone network and the homogeneous graph augmentation module (DA) shows significant improvement on the Amazon and YelpChi datasets, but not so well on the T-Finance and T-Social datasets. Since the Amazon and YelpChi datasets have at most tens of thousands of nodes, the neighboring nodes and edges are relatively few, so the operation of deleting noisy edges has a good promoting effect on the node representation. In contrast, T-Finance and T-Social have millions of edges, and our deletion of noisy edges only deletes a part of the edges, which may not significantly improve the node representation with millions of edges. However, this also proves the effectiveness of this method. Their AUROC, AUPRC and MacroF1 are shown in Figure 2 and Figure 3.

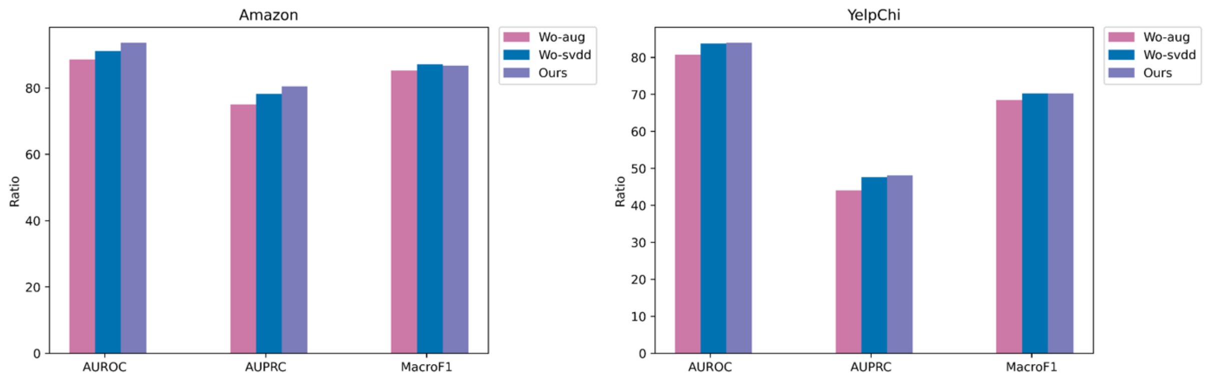

Figure 2. Ablation of Amazon and YelpChi(Wo-aug represents no edges deleted, Wo-svdd represents not using spherical loss.)

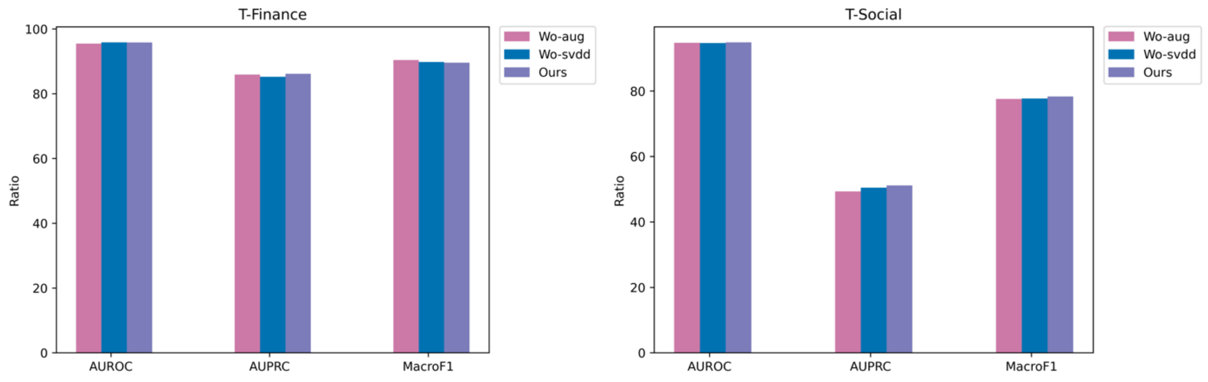

Figure 3. Ablation of T-Finance and T-Social(Wo-aug represents no edges deleted, Wo-svdd represents not using spherical loss.)

3.4.3. *Homogeneous graph augmentation.* To explore whether the homogeneous graph augmentation operation can improve the homogeneity rate of the graph during the learning process, this section selects the two datasets with the most edge relation types for comparison. Here, the homogeneity rate of edges is used to replace the homogeneity rate of nodes, as our node features are obtained based on edge embeddings. Meanwhile, our deletion ratio is set to delete noisy edges at a certain proportion for each type of edge relation. In Figure 4, there are two homogeneity rates: ori represents the homogeneity rate of the original graph, and aug represents the homogeneity rate of the graph after deleting an appropriate proportion of noisy edges according to the type of edge relation. It can be seen that on the YelpChi dataset, the homogeneity rate of each type of edge relation has been improved to some extent, while on the Amazon dataset, the improvement in homogeneity rate is not very obvious. This may be because Amazon has too many edges, and the homogeneity rate of Amazon itself is already very high. The impact of deleting noisy edges on the dataset is limited. However, this still proves that the operation of deleting noisy edges to augment the homogeneous graph is effective.

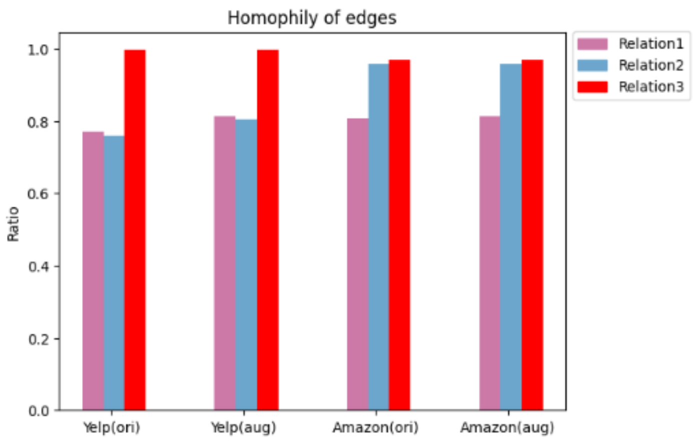

Figure 4. Homophily of edges(ori represents the original graph homophily of edges, aug represents the homophily of the graph after deleting some edges.)

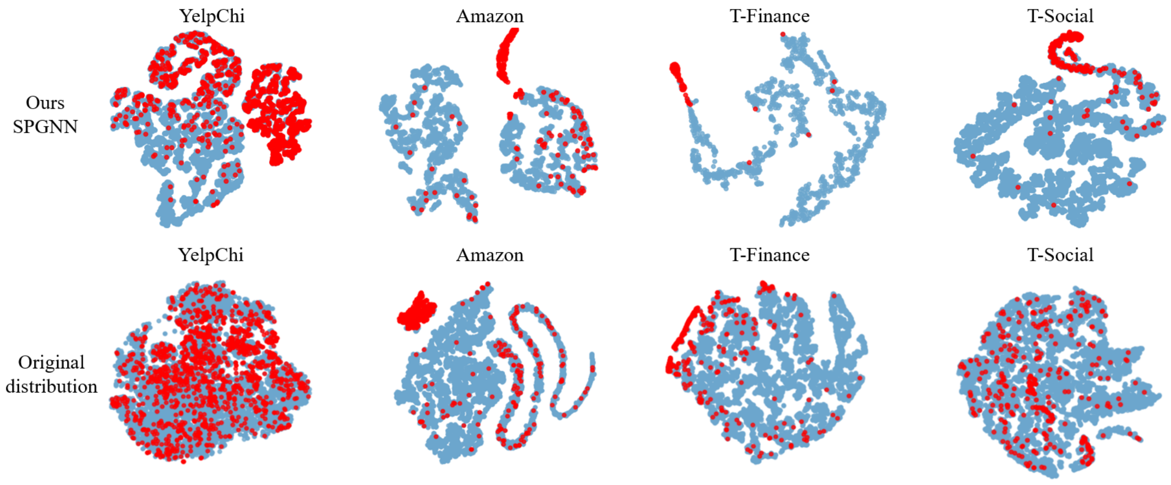

Figure 5. Feature space embedding visualization of SPGNN and original distribution

3.4.4. *The visualization of feature distribution.* To explore the quality of graph data feature embedding during the learning process, this section intuitively compares the sampling embeddings of different datasets. The number of randomly sampled samples is $10,000$ nodes. It can be clearly seen that in Figure 5, when the model is not used for extraction embedding, there is no obvious boundary between the feature embeddings of abnormal and normal nodes. However, in Figure 5, when the model is used for extraction embedding, there is an obvious boundary between the feature embeddings of abnormal and normal nodes. The sample embeddings of these four datasets are used as the input of the t-SNE tool to generate a two-dimensional visualization image of the sample embeddings. The results are shown in Figure 5, where blue corresponds to the normal category and orange corresponds to the abnormal category.

4. Conclusion.

In this paper, we have studied the problem of detecting abnormal nodes using graph data and proposed a method named SPGNN for detecting abnormal nodes under limited supervision. This method combines the graph neural network method in the spectral domain with the neural network method in the spatial domain. In the spectral domain, it deals with the problem of edge noise by deleting heterogeneous edges, and in the spatial domain, it addresses the over-smoothing problem through edge feature embedding. The experimental results have demonstrated that SPGNN outperforms traditional anomaly detection methods and can also exhibit excellent performance on datasets with millions of nodes, showing its potential for application in industrial datasets. For example, in the future, on Grab, a leading lifestyle platform in Southeast Asia, open data is collected to construct a graph, and then the trained model is saved in file form. An API interface is developed to call our model, which can analyze historical information on the Grab platform, detect fraudulent orders, and prevent user economic losses. In future work, we hope to explore a more effective way to combine graph signal processing methods and spatial-based message passing methods to construct feature processing methods for nodes. This will not only handle the over-smoothing phenomenon better but also expand the potential of the proposed method in industrial applications.

Acknowledgment. This work is supported by Henan Province Major Science and Technology Project (221100810100), and supported by Natural Science Foundation of Henan Province (242300420675).

REFERENCES

[1] W.-Z. Li, C.-D. Wang, H. Xiong, and J.-H. Lai, "Homogcl: Rethinking homophily in graph contrastive learning," in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023, pp. 1341–1352.

[2] D.-X. He, J.-T. Zhao, R. Guo, Z.-Y. Feng, D. Jin, Y.-X. Huang, Z. Wang, and W.-X. Zhang, "Contrastive learning meets homophily: two birds with one stone," in *International Conference on Machine Learning*. PMLR, 2023, pp. 12 775–12 789.

[3] S. K. Maurya, X. Liu, and T. Murata, "Improving graph neural networks with simple architecture design," *arXiv preprint arXiv:2105.07634*, 2021.

[4] J. Huang, P. Li, R. Huang, N. Chen, and A. Zhang, "Revisiting the role of heterophily in graph representation learning: An edge classification perspective," *ACM Transactions on Knowledge Discovery from Data*, vol. 18, no. 1, pp. 1–17, 2023.

[5] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger, "Simplifying graph convolutional networks," in *International Conference on Machine Learning*. PMLR, 2019, pp. 6861–6871.

[6] L. Cui, H. Seo, M. Tabar, F. Ma, S. Wang, and D. Lee, "Deterrent: Knowledge guided graph attention network for detecting healthcare misinformation," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 492–502.

[7] C. Liu, L. Sun, X. Ao, J. Feng, Q. He, and H. Yang, "Intention-aware heterogeneous graph attention networks for fraud transactions detection," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 3280–3288.

[8] M. Huang, Y. Liu, X. Ao, K. Li, J. Chi, J. Feng, H. Yang, and Q. He, "Auc-oriented graph neural network for fraud detection," in *Proceedings of the ACM Web Conference 2022*, 2022, pp. 1311–1321.

[9] T. Zhao, Y. Liu, L. Neves, O. Woodford, M. Jiang, and N. Shah, "Data augmentation for graph neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 12, 2021, pp. 11 015–11 023.

[10] L. Shi, L. Wang, C. Long, S. Zhou, M. Zhou, Z. Niu, and G. Hua, "Sgcn: Sparse graph convolution network for pedestrian trajectory prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8994–9003.

[11] M. He, Z. Wei, H. Xu *et al.*, "Bernnet: Learning arbitrary graph spectral filters via bernstein approximation," *Advances in Neural Information Processing Systems*, vol. 34, pp. 14 239–14 251, 2021.

[12] J. Tang, J. Li, Z. Gao, and J. Li, "Rethinking graph neural networks for anomaly detection," in *International Conference on Machine Learning*. PMLR, 2022, pp. 21 076–21 089.

[13] K. Lu, Y. Yu, H. Fei, X. Li, Z. Yang, Z. Guo, M. Liang, M. Yin, and T.-S. Chua, "Improving expressive power of spectral graph neural networks with eigenvalue correction," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 13, 2024, pp. 14 158–14 166.

[14] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in Neural Information Processing Systems*, vol. 30, pp. 1025–1035, 2017.

[15] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[16] Q. Wu, W. Zhao, C. Yang, H. Zhang, F. Nie, H. Jiang, Y. Bian, and J. Yan, "Sgformer: Simplifying and empowering transformers for large-graph representations," *Advances in Neural Information Processing Systems*, vol. 36, pp. 64 753–64 773, 2023.

[17] C. Yang, J. Liu, Y. Yan, and C. Shi, "Fairsin: Achieving fairness in graph neural networks through sensitive information neutralization," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 8, 2024, pp. 9241–9249.

[18] J. J. McAuley and J. Leskovec, "From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews," in *Proceedings of the 22nd International Conference on World Wide Web*, 2013, pp. 897–908.

[19] S. Rayana and L. Akoglu, "Collective opinion spam detection: Bridging review networks and metadata," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015, pp. 985–994.

[20] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain." *Psychological Review*, vol. 65, no. 6, p. 386, 1958.

[21] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[22] Z. Liu, Y. Dou, P. S. Yu, Y. Deng, and H. Peng, "Alleviating the inconsistency problem of applying graph neural network to fraud detection," in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020, pp. 1569–1572.

[23] Y. Liu, X. Ao, Z. Qin, J. Chi, J. Feng, H. Yang, and Q. He, "Pick and choose: a gnn-based imbalanced learning approach for fraud detection," in *Proceedings of the Web Conference 2021*, 2021, pp. 3168–3177.

[24] F. Shi, Y. Cao, Y. Shang, Y. Zhou, C. Zhou, and J. Wu, "H2-fdetector: A gnn-based fraud detector with homophilic and heterophilic connections," in *Proceedings of the ACM Web Conference 2022*, 2022, pp. 1486–1494.

[25] Y. Gao, X. Wang, X. He, Z. Liu, H. Feng, and Y. Zhang, "Alleviating structural distribution shift in graph anomaly detection," in *Proceedings of the sixteenth ACM International Conference on Web Search and Data Mining*, 2023, pp. 357–365.

[26] Y. Wang, J. Zhang, Z. Huang, W. Li, S. Feng, Z. Ma, Y. Sun, D. Yu, F. Dong, J. Jin *et al.*, "Label information enhanced fraud detection against low homophily in graphs," in *Proceedings of the ACM Web Conference 2023*, 2023, pp. 406–416.

[27] T.-Y. Wu, H. Li, S. Kumari, and C.-M. Chen, "A spectral convolutional neural network model based on adaptive fick's law for hyperspectral image classification," *Computers, Materials & Continua*, vol. 79, no. 1, 2024.