2025 International Conference on Intelligent Computing
July 26-29, Ningbo, China
https://www.ic-icc.cn/2025/index.php

HH-GNN: Homogeneity- and Heterogeneity-Aware Graph Neural Network for Fraud Detection with Noisy Labels

Boyi He[1], Jianzhe Zhao[1], Xuan Wang[2], Wei Ai[3] and Tao Meng[2]

[1] Northeastern University, Software College, Shenyang 110819, Liaoning, China
20226956@neu.edu.cn, zhaojz@swc.neu.edu.cn
[2] Hunan University, College of Finance and Statistics, Changsha 410079, Hunan, China
xuanw@hnu.edu.cn
[3] Central South University of Forestry and Technology, College of Computer and Mathematics, Changsha 410082, Hunan, China
aiwei@hnu.edu.cn, mengtao@hnu.edu.cn

Abstract. Because graph structures can represent rich information by aggregating neighborhood information, graph neural networks (GNNs) are heavily used in fraud detection tasks. However, a large amount of noise is generated in fraud detection problems that affect the detection effectiveness of the model. On the one hand, the fraudster actively generates noise through two disguises: feature disguise and relationship disguise; on the other hand, a part of the noise is also generated in the graph construction due to the fact that the labeling of the adopted data is not guaranteed to be correct as well as the connection between normal nodes and fraudulent nodes unconsciously. In order to address such problems, we propose a framework that focuses on both homogeneous and heterogeneous information (HH-GNN) in the paper. It improves the noise at graph nodes and connections by considering both homogeneous and heterogeneous information in the distance calculation method and the dilated k-NN algorithm to achieve neighbor aggregation. Meanwhile, based on the early learning phenomenon, we introduce ELR regularization to effectively suppress the influence of noisy labels during gradient descent. Our experiments on fraud detection tasks on four real datasets using multidimensional metrics of AUC value, and F1-macro show the effectiveness and superiority of the proposed HH-GNN.

Keywords: Fraud Detection, Graph Neural Networks, Node Classification.

1 Introduction

As the Internet advances rapidly, more and more fraudulent behavior began to appear, causing great economic losses. Fraud detection has become an important proposition that needs to be studied. Early fraud detection used shallow machine learning methods like support vector machines and decision trees [1][2], but the method itself ignores the connection information of nodes and neighbors and thus fails to achieve better detection

results. With the development of graph neural networks, the node detection problem is used in application areas such as social networks [3] or finance [4]. People are beginning to discover the contrasting nature of the interactions between nodes and nodes in fraud detection with nodes and edges in graph neural networks, and are applying node identification techniques to fraud detection tasks.

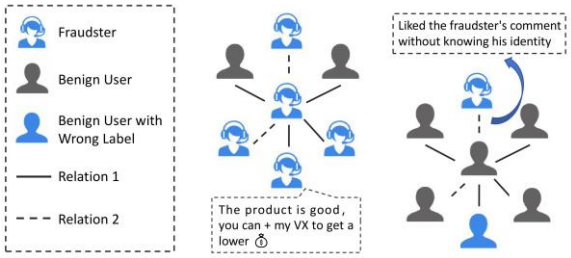

Fig. 1. A description of the two noises. (1) Fraudsters create noise by adding special symbols and connecting intentional and benign nodes. (2) Because errors in the label set and unintentional connections of benign nodes can lead to noise during graph construction.

Although there are a large number of studies using graph neural networks in fraud detection tasks, such as CARE-GNN [5], PC-GNN [6], they are based on the traditional assumption of homogeneity and the fact that fraudulent nodes in fraud detection tend to camouflage their relationships by making heterogeneous connections with their benign node neighbors. For instance, in the network of financial transactions, Fraudsters frequently leverage legitimate users to carry out transactions [7]. At the same time, fraudulent nodes will also change their node characteristics (e.g., fraudulent nodes will add special symbols to their comments to complete the camouflage) [5] so as to make them similar to ordinary nodes. The above two scenarios are common camouflage problems in fraud checking as such. The above camouflage behaviors will generate a lot of noisy information in specific detection tasks, affecting the effectiveness of the model.

Additionally, the data itself and the construction of the graph in real fraud detection scenarios will also generate a lot of noise. On the one hand, in reality, some benign node users will unknowingly link to some fraudulent nodes through actions such as clicking and commenting. This will generate noisy edges when the graph is constructed, causing it to be misjudged as a fraudulent node. On the other hand, because there is less manually annotated sample data that can be used for training [8], some training processes may use labels from data sources such as web pages to annotate the data [9]. This data will have some incorrect annotations, generating noisy labels that affect the training results. Therefore, solving the impact of noise on detection results is a new problem needed to be addressed in the fraud detection task.

Graph-based fraud detection faces two major challenges. **(1) How to deal with the noise problem caused by fraudulent node camouflage behavior.** Fraudulent nodes usually mitigate suspicions about themselves by changing their characteristics and connecting more often to benign nodes. Most of the methods perform pruning by calculating the resemblance between the target node and its adjacent neighbors, like CGDF-

GNN [10], these methods intensify the representation of the node's characteristics but do not take into account the effect of heterogeneous information on the target node. **(2) How to deal with the noise problem in graph construction and training caused by training data.** There are few existing datasets that can be used for fraud detection training, and they contain noisy information caused by user behavior. Existing methods [8] solve this problem by introducing feature decoupling and consistency regularization to enhance the learning of a limited amount of labeled data and a substantial amount of unlabeled samples. In other deep learning fields, non-manually labeled data is introduced, and regularization techniques are used to reduce the affect of noise information on training results [11], but this method has not yet been applied to fraud detection tasks.

To address the two challenges, we propose an innovative **H**omogeneity- and **H**eterogeneity-Aware **G**raph **N**eural **N**etwork for fraud detection in the presence of noisy labels (HH-GNN). For the disguise problem, we designed a new neighbor aggregation method. This method obtains the similarities and differences between the target node and the adjacent neighbors and completes the screening and aggregation of neighbor nodes to reduce the disguise of fraudulent nodes in terms of relationships and features. For the noisy label problem, we introduced an early-regularization meth-od. This approach guides the model toward these objectives, indirectly avoiding memorization of incorrect labels. The key contributions of our method, HH-GNN, can be outlined as below:

• We design a novel neighbor aggregation strategy that considers both homogeneous and heterogeneous information of a node and its neighbors to address the camouflage in fraud detection.

• We introduce an early-regularization method that addresses the noisy label problem in the dataset by preventing early false memories from affecting model training.

• We undertake extensive experimentation on four state-of-art benchmark datasets to prove the efficacy of HH-GNN compared to the most advanced methods.

2 Related Work

2.1 Semi-supervised Learning Based on Graph Structures

The task of semi-supervised node classification is to leverage labeled data to predict attributes of unlabeled nodes. Recently, graph neural networks have demonstrated amazing abilities in this area. There are two main types of graph neural networks: (1) Spectral-based GNNs, which convert graphs into Laplacian matrices to implement convolution operations in the spectral domain, for instance GCN [12]. (2) Spatial-based GNNs, which disseminate information through aggregation of neighbor nodes using spatial relationships, like GraphSAGE [14] and GAT [13]. However, this method is based on the assumption of similarity between the target node and its neighbors. The disguised behavior of fraudulent nodes in fraud detection tends to connect to benign nodes, which violates this assumption. Our method avoids the above problem by adjusting the graph structure based on neighbor information in advance.

2.2 Popular Graph-Based Fraud Detection Model

In recent years, graph neural networks have been used in a wide range of fraud detection because of their matching with various relationship nodes and node interaction patterns in fraud detection. GraphConsis [24] initially brought graph neural networks into the field of fraud detection， witch uses a predefined threshold to selectively sample adjacent nodes. After that, CARE-GNN [5] and CGDF-GNN [10] performed pruning by comparing the similarity between target nodes and neighboring nodes and adjusted the graph structure to obtain better node representation. This method has achieved certain results, but it ignores the role of heterogeneous information in node detection. GHRN [15] and SEC-GDF [16] obtain and utilize heterogeneous information in the graph by integrating high-frequency and mixed filters. The above methods are all trained on fixed manually annotated datasets, which cannot adapt to the situation where there are fewer datasets for fraud detection tasks and the cost of manual annotation is high. Another method [8] designs a barely supervised learning approach using feature disentanglement and consistency regularization to improve fraud detection performance with limited labeled samples. However, this method is not considered the most effective method of training the model on datasets containing noisy labels. Our method achieves model training on noisy labeled datasets by introducing early regularization.

2.3 Model predictions for noisy labels

As the demand for datasets increases, a lot of data with noisy annotations are generated to train deep learning classification models. The following are the most widely used methods: (1) Loss-correction methods. This method mainly takes into account the noise distribution by explicitly correcting the loss function. The noise distribution is represented by the transfer matrix of the probability of incorrect labeling [17]. (2) Label Correction Methods. This method uses the characteristic that "in the early learning stage, the model's prediction of some incorrect labels may be more accurate" to replace or correct these incorrect labels with pseudo-labels, such as soft labels, hard labels [18], or mixed labels [19] to iteratively correct the labels of noisy samples. (3) Consistency Regularization. Regularization itself plays a significant role in the recognition of noisy labels. It improves the model's resilience and capacity to generalize by maintaining the consistency of the model's output under different data augmentations [9]. Our method combines the characteristics of the above methods, calculates a probability estimate similar to the above soft label, and then uses it to avoid memorization. Then, a novel regularization term is used to explicitly adjust the gradient of the cross-entropy loss function [11].

3 Method

This section present the overall framework of our proposed model, HH-GNN, and provide a comprehensive explanation of its methodological details. As illustrated in Fig. 2, our framework is composed of three key components: (1) a neighbor filter, which

effectively captures both the similarities and differences among neighboring nodes; (2) a neighbor information aggregator that integrates relevant information; and (3) a gradient descent module, designed to alleviate the adverse effects of noisy labels, thereby enhancing model robustness and performance.

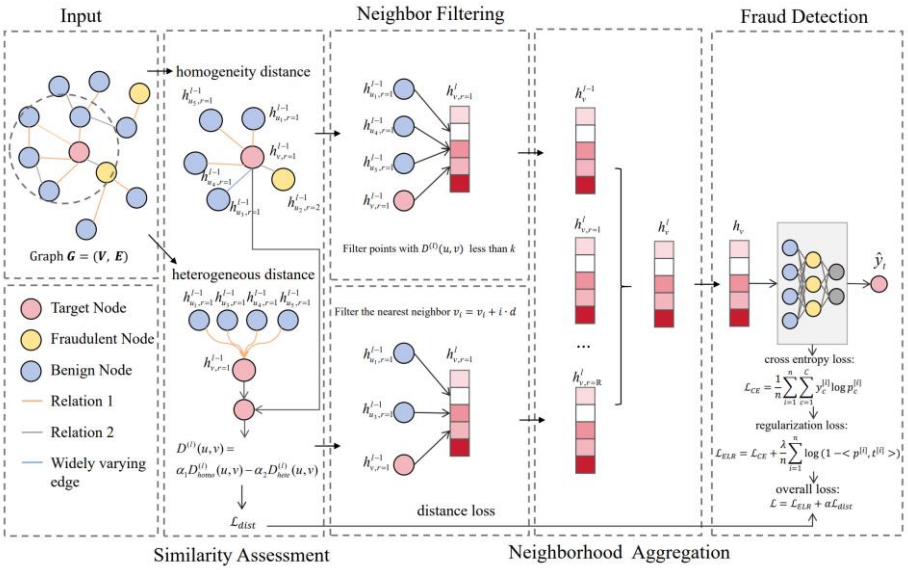

Fig. 2. The overall process of the proposed HH-GNN. It includes a detailed demonstration of the framework's operation including: Similarity Assessment, Neighbor Filtering and Aggregation, Fraud Detection.

3.1 Problem Definition

We define a multi-relation graph as $G = \{\mathcal{V}, \mathcal{E}, X\}$,where $\mathcal{V} = \{v_1, \cdots, v_N\}$ represents its node set. $\mathcal{E}$ denotes the edge set, while $A_{ij} = 1$ indicates an edge between nodes $\mathrm{v_i}$ and $\mathrm{v_j}$, where $A \in \{0,1\}^{N\times N}$ denotes the adjacency matrix of G. The node hallmark set is represented as $X = \{x_1, \ldots, x_N\} \in R^{N\times d}$, where each node $\mathrm{v_n}$ has at most d-dimensional feature vector. $Y = \{y_1, \ldots, y_N\}$ represents node labels, where $y_n(0/1)$ is the ground-truth label, with 0 for benign nodes and 1 for fraudsters.

Refraining our perspective, we view graph-based fraud detection as a semi-supervised binary classification task at the node level, distinguishing nodes in the fraud graph into labeled and unlabeled groups. The labeled nodes are represented as Y_{train}, whereas the labels for the unlabeled nodes, Y_{test}, are not visible during training. Therefore, our aim is to derive a function for assigning anomaly labels to the unlabeled nodes, utilizing all available information that we possess:

$$Y_{\text{test}} = f(X, A, Y_{\text{train}}) \tag{1}$$

3.2 Neighbor similarity assessment

Because fraudsters often have disguised features in fraud detection scenarios, they usually disguise themselves by disguising node features or choosing to connect to benign nodes. This usually produces incorrect node embeddings, which in turn misleads the final node classification and leads to recognition errors. The existing method mainly considers the homogeneity information of nodes to calculate similarity [5][10], which cannot completely solve the problem. Therefore, we use a more comprehensive method that considers both homogeneity and heterogeneity to calculate similarity.

We first calculate the distance of nodes on homogeneity. Inspired by CARE-GNN [5], we employ a parameterized similarity metric to assess the distance between nodes. It integrates a fully connected network (FCN) with linear regularization and uses the l_1 −distance between the prediction results of two nodes as the similarity metric. The node distance in the homogeneity dimension is defined as bellow:

$$\mathcal{D}_b^{(\ell)}(u,v) = \left\|\sigma\left(FCN^{(\ell)}\left(\mathrm{h}_u^{(\ell-1)}\right)\right)\right\| - \left\|\sigma\left(FCN^{(\ell)}\left(\mathrm{h}_v^{(G)}\right)\right)\right\| \tag{2}$$

where $\mathrm{h}_v^{(G)}$ represents the original embedding of node *v* and σ () represent the nonlinear activation function, for which we adopt tanh in our approach. At the same time, we introduce a node distance measure based on the heterogeneity dimension. We measure the heterogeneous distance of nodes by calculating the difference in hallmark between nodes and removing the influence of node degrees:

$$\mathcal{D}_{\mathrm{hete}}^{(\ell)}(u,v) = \frac{\left\|\mathrm{h}_u^{(\ell-1)} - \mathrm{h}_v^{(G)}\right\|}{\sqrt{d_r(u)d_r(v)}} \tag{3}$$

where $d_r(u)$ and $d_r(v)$ represents the degree of node *u*, *v*, respectively. And, in order to combine homogeneity and heterogeneity information, we combine the two distance formulas to obtain the final similarity measurement formula. At the same time, due to the opposite contributions of the two to the similarity, we use the two parameters α_1 and α_2 as learnable attention weights. We get the final similarity formula as follows:

$$\mathcal{D}^{(l)}(u,v) = \alpha_1 \mathcal{D}_{homo}^{(l)}(u,v) - \alpha_2 \mathcal{D}_{hete}^{(l)}(u,v) \tag{4}$$

To better evaluate the similarity distance between nodes, we use direct supervision signals from the labels to train the similarity metric. Under a single relationship, we apply the cross-entropy loss to optimize this distance function:

$$\mathcal{L}_{\mathrm{dist}} = -\sum_{l=1}^{L}\sum_{v\in\mathcal{V}}\left[y_v \log\left(p_v^{(l)}\right) + (1-y_v)\log\left(1-p_v^{(l)}\right)\right]p_v^{(l)} \tag{5}$$

3.3 Neighbor Aggregation

After obtaining the similarity-based distance between the target node and its surrounding neighbors, we need to re-obtain its neighbor subgraph *G* through the sampling module. Then we re-aggregate the neighbor node features on the subgraph *G* to obtain a

new feature embedding representation to decline the impact of feature camouflage on the model results.

Most Similar Neighbor Filtering. For the target node, we obtain the similarity between it and its neighbor nodes through the distance function and filter it through formula 6 to obtain a new subgraph $\mathcal{N}_r^{(l)}(u)$ (the graph contains the target node and the similar neighbor nodes after filtering):

$$\mathcal{N}_r^{(l)}(u) = \{v \in \mathcal{V} \mid \mathcal{A}_r(u,v) > 0 \text{and} \mathcal{D}^{(l)}(u,v) < k\} \tag{6}$$

where $\mathcal{D}^{(l)}(u,v)$ is obtained from formula 5, $r \in \{1,\dots,R\}$, l denotes the number of layers, while *k* is a hyperparameter that defines the filtering threshold.

Neighbor Filtering Using the Dilated k-NN Algorithm. With the increasement of the number of network layers, if we blindly select the most similar neighbor nodes for aggregation, the problem of over-smoothing may easily occur. To dispatch this problem and expand the receptive area in traditional graph convolutional networks, we use the dilated *k-NN* algorithm to dynamically search the *k* neighbor nodes of node *u* to form a neighbor subgraph $\mathcal{N}_{\mathrm{r}}^{(l)}(\mathrm{u})$ of the target node before aggregating neighbor information. Different from selecting the *k* neighbor nodes with the highest similarity, this method searches for *k* neighbor nodes that are relatively similar to the target node by selecting similar neighbor nodes and then skipping *d* neighbors, where *d* represents the dilation rate. Therefore, the *k* neighbors of node *u* with dilation rate *d* can be represented as:

$$\mathcal{N}_{k-NN}^{(l)}(u) = \{v_i \mid \mathcal{A}_r(u,v_i) > 0 \text{and } v_i = v_{1+i\cdot d}, i = 0,1,\dots,k-1\} \tag{7}$$

Neighborhood feature aggregation. Once each node's neighbors are filtered, the subsequent step involves aggregating information from its surrounding nodes under various relationships. This process can be broken down into two stages: aggregation and combination. First, we will obtain the feature embeddings of the neighbor nodes and aggregate them into the embedding of the target node:

$$\mathrm{h}_{v,r}^{(l)} = \mathrm{ReLU}\left(W_r^{(l)}\left(\mathrm{h}_{v,r}^{(l-1)} \oplus AGG_r^{(l)}\left\{\mathrm{h}_{u,r}^{(l-1)}\right\}\right)\right) \tag{8}$$

where $AGG_r^{(l)}$ is the aggregation function based on averaging at the *l*-th level under relation *r*, $\oplus$ represents the connection operation, $W_r^{(l)} \in \mathbb{R}^{d_l \times 2d_{l-1}}$ denotes the weight matrix, while $v \in \mathcal{N}_r^{(l)}(u)$. Then, we combine the node embedding of the previous layer and the embedding of each relationship in this layer to get the node's embedding under this layer:

$$\mathrm{h}_v^{(l)} = \mathrm{ReLU}\left(W^{(l)}\left(\mathrm{h}_v^{(l-1)} \oplus \mathrm{h}_{v,r=1}^{(l)} \oplus \cdots \oplus \mathrm{h}_{v,r=R}^{(l)}\right)\right) \tag{9}$$

where $W_r^{(l)} \in \mathbb{R}^{d_l \times (d_{l-1}+R \cdot d_l)}$ is the weight matrix.

3.4 Fraud Detection and Node Noise Improvement

Fraud Detection. After obtaining the final embedding h_{v_t} of the target node, we carefully designed the objective function of GDF-ELR. We input the embedding h_{v_t} into the final classification function to infer the label associated with the target node v_t:

$$\hat{y}_t = \psi\left(W_f^T \cdot h_{v_t} + b_f\right) \tag{10}$$

where $W_f \in \mathbb{R}^{3U}$ denotes the learnable weight matrix, while $b_f \in \mathbb{R}^{3U}$ represents the bias vector. ψ represents the activation function $softmax$.

Node Noise Improvement. Considering that the fraud nodes in the fraud detection task generate noisy labels by disguising and calling crowdsourced data will also have noisy label problems [9], we establish a suitable loss function to minimize the impact of noisy labels on training results. Among them, changing the regularization term of the loss function to achieve better training results in datasets with noisy labels is a common method in semi-supervised learning. Previous studies have found that when training on noisy labels, in the early learning stage, deep neural networks will initially fit the training data with accurate labels, and then eventually memorize examples with incorrect labels [20]. In response to the above phenomenon, we can achieve robustness to noisy labels by adding or changing the idea of regularization. ELR [11] attempts to use the above early memory phenomenon to improve the prediction accuracy of classification models under noisy labels.

Based on the above methods, we devise a suitable regularization term to strengthen the influence of early clean labels and offset the influence of noisy labels. Our method introduces the model's prediction probability $p^{[i]}$ and target probability $t^{[i]}$, where $t^{[i]}$ is calculated using the temporal ensembling [21] technique in class interpretation learning. Let $t^{[i]}(k)$ and $p^{[i]}(k)$ represents the target and the model's output, respectively, for instance, *i* at the *k*-th training iteration. We define:

$$\mathbf{t}^{[i]}(k) := \beta \mathbf{t}^{[i]}(k-1) + (1-\beta)\mathbf{p}^{[i]}(k) \tag{11}$$

where $0 \leq \beta < 1$ denotes the momentum. Because of the early-learning effect, we presume that during the initial stages of the optimization, the targets will not become overly fitted to the noisy labels. To address this, we use a regularization component designed to maximize the alignment between predicted outputs and reference targets via their inner product.

$$\mathcal{L}_{\mathrm{CE}} = \frac{1}{n}\sum_{i=1}^{n}\sum_{c=1}^{C} y_c^{[i]} \log p_c^{[i]} \tag{12}$$

$$\mathcal{L}_{\mathrm{ELR}} = \mathcal{L}_{\mathrm{CE}} + \frac{\lambda}{n}\sum_{i=1}^{n}\log\left(1 - \left\langle \mathbf{p}^{[i]}, \mathbf{t}^{[i]} \right\rangle\right) \tag{13}$$

During gradient descent, for $\mathcal{L}_{\mathrm{CE}}$, the noise label $y^{[i]}$ moves in the opposite direction of $x^{[i]}$ and produces the phenomenon of memorization. In gradient descent with noisy labels, since the cross entropy $p^{[i]} - y^{[i]}$ in the clean labels will tend to 0, this makes the noisy labels dominate in the later training, affecting the robustness of the model. The regular term added in $\mathcal{L}_{\mathrm{ELR}}$ can solve this problem. In gradient descent, for clean labels, it will add a positive parameter $g^{[i]}top^{[i]} - y^{[i]}$ to amplify its effect on the results, and on the contrary, for noisy label, *s* it will add a negative parameter $-g^{[i]}$ to counteract its effect to reduce the memorization phenomenon.

Then, to consider the impact exerted by the distance function, we introduce the loss function $\mathcal{L}_{\mathrm{dist}}$ of the distance function into the final loss function, where α functions as a parameter to strike a balance between optimizing the distance function and the classifier.

$$\mathcal{L} = \mathcal{L}_{\mathrm{ELR}} + \alpha\mathcal{L}_{\mathrm{dist}} \tag{14}$$

4 Experiments

This section presents a comprehensive empirical evaluation of the method we propose using commonly adopted benchmark datasets. By performing a comprehensive set of experiments, we systematically compare its performance with state-of-the-art models. These comparisons highlight the effectiveness of our method, demonstrating its superiority in tackling the given fraud detection task.

4.1 Experimental Settings

Datasets. We conducted extensive experiments on four real-world datasets for fraud detection: Amazon [22], focusing on music instrument reviews, and YelpChi [23], containing hotel and restaurant reviews from Yelp. Both datasets include three types of node relationships. In addition, we use two updated datasets from the real world, T-Finance and T-Social [25], which detect abnormal users by capturing information about each account in a social network and the relationships of accounts that have transactions in them. Both of these datasets have only one connection relationship, while the T-Social dataset has 100 times more data than the Amazon and Yelp datasets.

Baseline. We evaluate the efficacy of our approach by comparing its performance against several GNN models designed for homophilic graphs, including GCN [12], GAT [13], and GraphSAGE [14]. Additionally, we benchmark our method against specialized graph-based fraud detection models, such as PC-GNN [6], CARE-GNN [5], GraphConsis [24], and BWGNN [25], to further assess its performance.

Evaluation Metrics. Due to the more serious sample imbalance problem in midterm detection (especially in the Yelp dataset) and our greater focus on fraudster detection results, we, therefore, selected two popular metrics in fraud detection to evaluate the overall performance across models: AUC and F1-Macro, where AUC evaluates the order of predicted probability of all instances that indicates the discriminative power of the model, while macro-F1 evaluates the model performance across different categories by considering the weighted average F1 scores.

Implementation. All baseline models are implemented using the official source code released by their respective authors to ensure fairness and reproducibility. In our proposed HH-GNN approach, we configure the final node embedding dimensionality to 64. For dataset-specific settings, we assign a batch size of 1024 for the T-Social, T-finance, and Yelp datasets and 256 for the Amazon dataset. Additionally, we train our model for 30 epochs and employ the Adam algorithm for efficient optimization, setting the learning rate as 0.01. To ensure consistency, all experiments are conducted in a Python 3.11 environment, leveraging appropriate computational resources to achieve optimal performance and reliable evaluation.

Table 1. The performance of fraud detection if evaluated across four datasets derived from real-world scenarios, utilizing both the F1-marco score and AUC value as metrics. (The bold values denote the best and runner-up performance.)

Type	Methods	Datasets			
		YelpChi	Amazon	T-Finance	T-Social
		AUC F1-Marco	AUC F1-Marco	AUC F1-Marco	AUC F1-Marco
General GNNs	GCN	54.36 51.52	74.07 64.67	64.43 70.74	84.35 59.98
	GAT	56.24 48.79	75.26 64.50	73.00 53.68	87.02 67.56
	GraphSage	56.45 46.10	75.27 64.64	67.12 52.71	70.80 59.77
Spatial GFD	PC-GNN	79.15 63.25	94.22 89.56	90.16 63.18	68.45 52.17
	CARE-GNN	77.72 60.89	87.28 88.34	90.66 77.36	71.86 56.26
	GraphConsis	75.69 65.58	87.03 78.25	90.22 63.18	68.45 52.17
	BWGNN	84.23 70.88	96.42 90.77	92.66 84.93	93.20 82.07
Ours	HH-GNN	85.48 72.350	98.53 92.97	93.00 85.34	92.97 83.89

4.2 Experimental Results

Overall performance. shows the performance of the model HH-GNN as well as the other baseline models on the four datasets. The results illustrate that our method has an outstanding performance under each metric, which is better than other methods. From the data in the table, we see that our model outperforms other models on all four datasets and compares favorably with state-of-the-art fraud detection models across all datasets. Among them, the AUC values in all datasets are more than 85, indicating the superior performance of our model.

In addition, our model showed better results compared to both types of baseline models. For traditional GNN models such as GAT, GCN, and GraphSage, which mostly ignore the noise that will be generated in various aspects of fraud detection, our model employs a unique convolutional process of neighbor screening to solve the problem. For advanced image-based fraud detection methods (e.g., PC-GNN), there is still a gap with our experimental results, which is largely due to the following two reasons: most of the above methods only consider homogeneous information among neighbors, while we consider heterogeneous information among neighbors; all of the above methods use the more traditional regularization for gradient descent, while our method introduces the ELR regularization in the gradient descent process, reducing the impact of noise and enhance the robustness of the model.

Effectiveness of Each Component of HH-GNN. In order to compare the effectiveness of top-k and k-NN methods in neighbor node screening and the validity of ELR regularity, we conducted ablation experiments. The following variants were tested: HH-GNN\top-k (top-k method in neighbor screening), HH-GNN\ k-NN (k-NN method in neighbor screening), and HH-GNN\ E (ELR regularity is used only without gradient descent), and the experimental outcomes are illustrated in Fig. 3.

For the different neighbor screening methods, the results in Fig. 3 (a) illustrate that the k-NN method achieves better results than the topK method. The k-NN method adopts an absolute threshold that cannot produce a suitable threshold for all the neighbor screening cases and, therefore, produces relatively poor results. The topK method adopts an interval screening method that can eliminate the overfitting phenomenon caused by too much similarity information to a certain extent, and achieves better experimental results. For the effectiveness of the ELR regularity, Fig. 3 (b) demonstrates that the regularity achieves a certain improvement compared to the general regularity, especially for the recall rate, which is very suitable for the needs of fraud detection that puts more emphasis on finding damaging fraudulent nodes that are damaging. Meanwhile, in the training, we found that the convergence speed of the model is significantly slower after adding the ELR regularizer, which is due to the fact that the regularizer requires a certain number of training rounds to complete the memorization of the early learning effect.

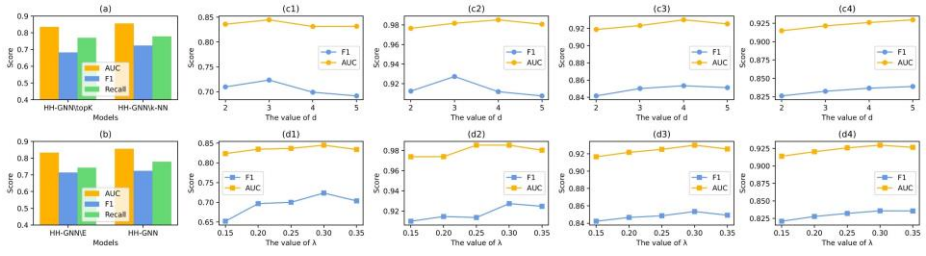

Fig. 3. (a) On the comparison of model effects using topK and k-NN methods in the neighbor screening stage. (b) Comparison of ablation experiments on whether or not to use ELR regularization in the gradient descent phase. (c) Sensitivity analysis of the spacing parameter *d* in the k-NN neighbor screening method. (d) Sensitivity analysis for the ELR canonical loss weight parameter λ in gradient descent.

Sensitivity to Hyper-Parameters. We further analyzed how the parameters affect the sensitivity of the HH-GNN model. Firstly, we examined the effect of different screening intervals *d* in the k-NNN nearest-neighbor screening method on the effectiveness of the model by gradually expanding the interval *d* from 2 to 5 and obtaining the experimental results. Figure 3 (c) shows that the experimental results are best when the interval *d* is 3, with the best performance in the two indices of F1 score and AUC value, while the model performance decreases after *d* is enlarged. From this, it can be obtained that appropriately enlarging the screening interval can avoid the influence of overfitting on the experimental results, while the interval is too large when it cannot effectively obtain the information of the nearest neighbors, resulting in the decline of the model performance. In addition, we also examined the effect of the loss weight λ in regular form on the model performance and obtained and compared the results by increasing λ continuously from 0.15 to 0.35. Figure 3 (d) demonstrates the corresponding results, which show that the model reaches its best results when λ is 0.3, while the convergence rate of the model tends to decrease as the value of λ increases.

5 Conclusions

In this paper, we propose a new framework that effectively realizes the fraud detection task. The framework uses a novel aggregation strategy and an early learning regularization formula that can effectively reduce the effect of noise on model training and detection results. A distance calculation formula that simultaneously considers the homogeneity and heterogeneity of nodes and neighbors is used in the aggregation process, while nearest-neighbor filtering and k-NN neighbor filtering are used to achieve aggregation in the aggregation process. An early learning regularization formula is introduced in gradient descent based on the early memory phenomenon to decline the impact of memory noise on the results. Experimental findings indicate that HH-GNN achieves superior performance compared to the latest models, thus validating the efficacy of our proposed aggregation method. However, the model still has problems such as poor early

convergence and long convergence time due to the learning memory process, waiting for us to find a suitable way to solve it in the next stage.

Acknowledgments. This research received partial support from the National Natural Science Foundation of China (Grant 69189338), the Excellent Young Scholars Program of Hunan Province (Grant 22B0275), and the Changsha Natural Science Foundation (Grant kq2202294).

Disclosure of Interests. The authors declare no conflicts of interest related to the content of this article.

References

1. Hooi B, Song H A, Beutel A, et al: Fraudar: Bounding graph fraud in the face of camouflage. In the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, 895–904 (2016) Author, F., Author, S.: Title of a proceedings paper. In: Editor, F., Editor, S. (eds.) CONFERENCE 2016, LNCS, vol. 9999, pp. 1–13. Springer, Heidelberg (2016)
2. Phua C, Lee V, Smith K, Gayler R: A comprehensive survey of data mining-based fraud detection research. arXiv preprint arXiv:1009.6119 (2010) Author, F.: Contribution title. In: 9th International Proceedings on Proceedings, pp. 1–2. Publisher, Location (2010)
3. Chen L, Peng J, Liu Y: Phishing Scams Detection in Ethereum Transaction Network. *ACM Trans. Internet Technol.*, **21**(1):10–25 (2020)
4. Zhang G, Li Z, Huang J, Wu J, Yang J, et al: eFraudCom: An E-commerce Fraud Detection System via Competitive Graph Neural Networks. *ACM Trans. Inf. Syst.*, **40**(3):47–75 (2022)
5. Dou Y, Liu Z, Sun L, et al: Enhancing graph neural network-based fraud detectors against camouflaged fraudsters, In *the 29th ACM international conference on information & knowledge management*, 315-324 (2020)
6. Y Liu, X Ao, Z Qin, et al: Pick and Choose: A GNN-based Imbalanced Learning Approach for Fraud Detection. Proceedings of the Web Conference 2021, 3168–3177 (2021)
7. Akoglu L, Tong H, Koutra D: Graph based anomaly detection and description: a survey. *Data Min. Knowl. Discovery*, 29(3): 626—688 (2015)
8. Yu H, Liu Z, Luo X: Barely Supervised Learning for Graph-Based Fraud Detection. Proceedings of the AAAI Conference on Artificial Intelligence, 38(15): 16548–16557 (2024)
9. Iscen A, Valmadre J, Arnab A, Schmid C: Learning with neighbor consistency for noisy labels in *the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4672–4681 (2022)
10. Wan Q, Wang P, Pei X: CGDF-GNN: Cascaded GNN fraud detector with dual features facing imbalanced graphs with camouflaged fraudsters. In *2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communications*, 258–265 (2022)
11. Liu S, Niles-Weed J, Razavian N, Fernandez-Granda C: Early-learning regularization prevents memorization of noisy labels. Advances in neural information processing systems, **33**: 20331–20342 (2020)
12. Kipf T N and Welling M: Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations* (2017)
13. P Velick´ović, G Cucurull, A Casanova, et al: Graph Attention Networks. In *International Conference on Learning Representations* (2018)
14. Hamilton WL, Ying R, and Leskovec J: Inductive Representation Learning on Large Graphs. In *Advances in Neural Information Processing Systems 30*, 1024— 1034 (2017)

15. Gao Y, Wang X, He X, et al: Addressing heterophily in graph anomaly detection: A perspective of graph spectrum. Proceedings of the ACM Web Conference 2023, 1528–1538 (2023)
16. Xu F, Wang N, Wu H, et al: Revisiting graph-based fraud detection in sight of heterophily and spectrum. Proceedings of the AAAI Conference on Artificial Intelligence, **38**(8):9214–9222, (2024)
17. Patrini G, Rozza A, Krishna Menon A, Nock R, and Qu L: Making deep neural networks robust to label noise: A loss correction approach. In Proc. IEEE Conf. Comput. Vis. Pattern Recognit.(CVPR), 2233—2241 (2017)
18. Tanaka D, Ikami D, Yamasaki T, and Aizawa K: Joint optimization framework for learning with noisy labels. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5552—5560 (2018)
19. Reed S, Lee H, Anguelov D, et al: Training deep neural networks on noisy labels with bootstrapping. In *CoRR*, (2014)
20. Devansh A, Stanisław J, Nicolas B, et al: A closer look at memorization in deep networks. In *the 34th International Conference on Machine Learning-Volume 70*, 233—242 (2017)
21. Samuli Laine and Timo Aila: Temporal ensembling for semi-supervised learning. In *ICLR*, (2018)
22. JJ 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*, 897–908 (2013)
23. 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*, 985–994 (2015)
24. Liu Z, Dou Y, Yu P S, et al: 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*, 1569–1572 (2020)
25. Tang J, Li J, Gao Z, et al: Rethinking graph neural networks for anomaly detection. Tn *International conference on machine learning*, PMLR, 21076–21089 (2022)