Article | Received 10 March 2025; Accepted 16 April 2025; Published 25 April 2024
https://doi.org/10.55092/aias20250004

Global confidence degree based graph neural network for financial fraud detection

Jiaxun Liu†, Yue Tian† and Guanjun Liu*

The Department of Computer Science, Tongji University, Shanghai 201804, China

† These authors contributed equally.

* Correspondence author; E-mail: {ljx0316, 1810861, liuguanjun}@tongji.edu.cn.

Highlights:

- Enhance separability between fraudulent/benign nodes via denoising feature transformation.
- Introduce Global Confidence Degree to quantify node typicality against global prototypes
- Achieve the simultaneous aggregation of both typical and atypical neighbor information.

Abstract: Graph Neural Networks (GNNs) are widely used in financial fraud detection due to their excellent ability on handling graph-structured financial data and modeling multilayer connections by aggregating information of neighbors. However, these GNN-based methods focus on extracting neighbor-level information but neglect a global perspective. This paper presents the concept and calculation formula of Global Confidence Degree (GCD) and thus designs GCD-based GNN (GCD-GNN) that can address the challenges of camouflage in fraudulent activities and thus can capture more global information. To obtain a precise GCD for each node, we use a multilayer perceptron to transform features and then the new features and the corresponding prototype are used to eliminate unnecessary information. The GCD of a node evaluates the typicality of the node and thus we can leverage GCD to generate attention values for message aggregation. This process is carried out through both the original GCD and its inverse, allowing us to capture both the typical neighbors with high GCD and the atypical ones with low GCD. Extensive experiments on two public datasets demonstrate that GCD-GNN outperforms state-of-the-art baselines, highlighting the effectiveness of GCD. We also design a lightweight GCD-GNN (GCD-GNN$_{light}$) that also outperforms the baselines but is slightly weaker than GCD-GNN on fraud detection performance. However, GCD-GNN$_{light}$ obviously outperforms GCD-GNN on convergence and inference speed.

Keywords: Graph Neural Networks (GNNs); financial fraud detection; prototype learning; graph anomaly detection

1. Introduction

Financial fraud is widespread and damaging, affecting both organizations and individuals. Economic scholars estimate that approximately 14.5% of large U.S. public companies engage in financial fraud, leading to an estimated 3% loss in enterprise value [1]. Large-scale corporations, including Enron in 2001, Wirecard in 2019, and Evergrande in 2021, have faced significant consequences due to these scandals. On a personal level, the increasing transaction frequency associated with various payment methods complicates oversight [2]. Therefore, detecting financial fraud is crucial to preventing substantial losses.

GNNs are widely used for mining structural data in financial fraud detection. Traditional GNNs often underperform due to the inherent characteristics of financial fraud activities, which include complex relationships and camouflage activities. (1) Complex relationships [3]: It is challenging to directly identify the relationships between entities based solely on their connections. (2) Camouflage activities [4]: Fraudsters employ strategies to obscure their fraudulent activities, thereby complicating detection. To address these challenges, some advanced GNN models employ attention mechanisms to assess the significance of edges during the message-passing process [5,6]. Other models focus on enhancing homophilous connections while reducing heterophilous ones [4,7]. In addition, some models use the label information to handle nodes in different categories separately [8,9]. All of these studies analyze financial fraud detection at the level of individual nodes and their neighbors. However, these methods overlook that neighbor messages can be harmful due to not only heterophily but also deceptive features, such as a fraudulent node camouflaged with normal features. This issue can be addressed on a global scale by evaluating the typicality of each node and eliminating messages based on their typicality, which aids in accurate classification, as demonstrated in unsupervised anomaly detection [10,11].

To fill the above gap, our paper aims to address financial fraud detection from global scale. Inspired by [12,13], we use the prototype to represent the global feature of a graph as global information. In our task, we seperately define two prototypes that are generated from all nodes in the same category. Following this, the article can address the following challenges: (1) How to generate an appropriate prototype to represent all nodes in a graph? The prototype should encapsulate the maximum amount of information from nodes within the same category, with each node contributing appropriately to its corresponding prototype. Moreover, unnecessary information should be eliminated to avoid overfitting. (2) How to extract Global Confidence Degree (GCD) for each node in a graph? We define the similarity between the prototype and each node as GCD to represent the typicality of a node. For labeled nodes, we can directly compare them with the prototype in the same category. For unlabeled nodes, we experiment with several methods to generate GCD and identify the approach that offers high performance and low time complexity. (3) How to utilize GCD in message generation? It is natural to maximize the extraction of the most typical information. However, atypical nodes (e.g., a node with features significantly different from its prototype) also provide valuable information.

To tackle the above issues, we propose a Global Confidence Degree Based Graph Neural Network (GCD-GNN). Firstly, we project the original features into a new space for extracting prototypes from these features which are then combined with the original features to be used for classification. Secondly, we propose a comparasion module to generate the GCD of each node. Thirdly, we utilize GCD to calculate

weight values for aggregation. In order to utilize both typical and atypical information, We aggregate messages from the typical and atypical perspectives separately as illustrated in Figure1. Inspired by [14,9], we employ a transformation matrix generated from the node's intrinsic features, as a component for message aggregation, ensuring that the node's own information directly influences the aggregation process.

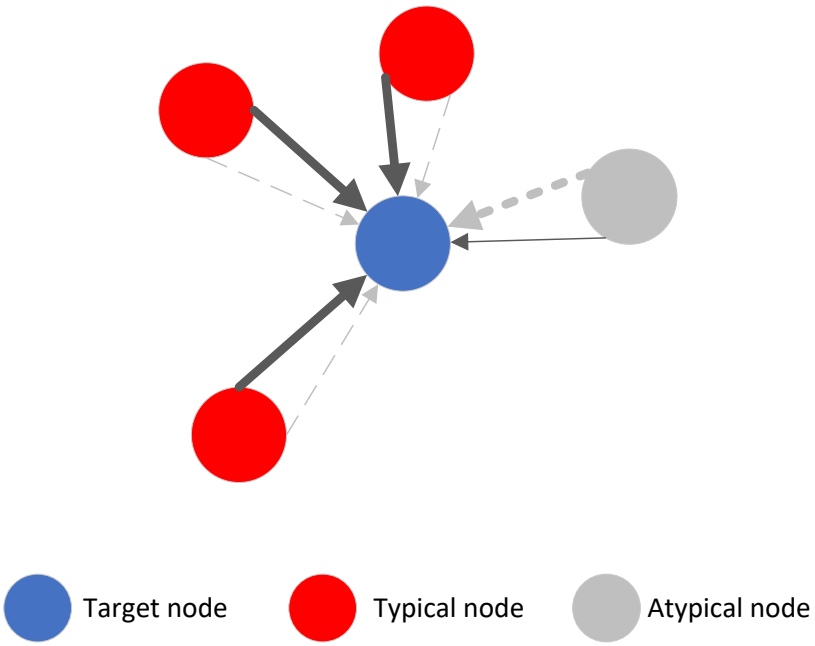


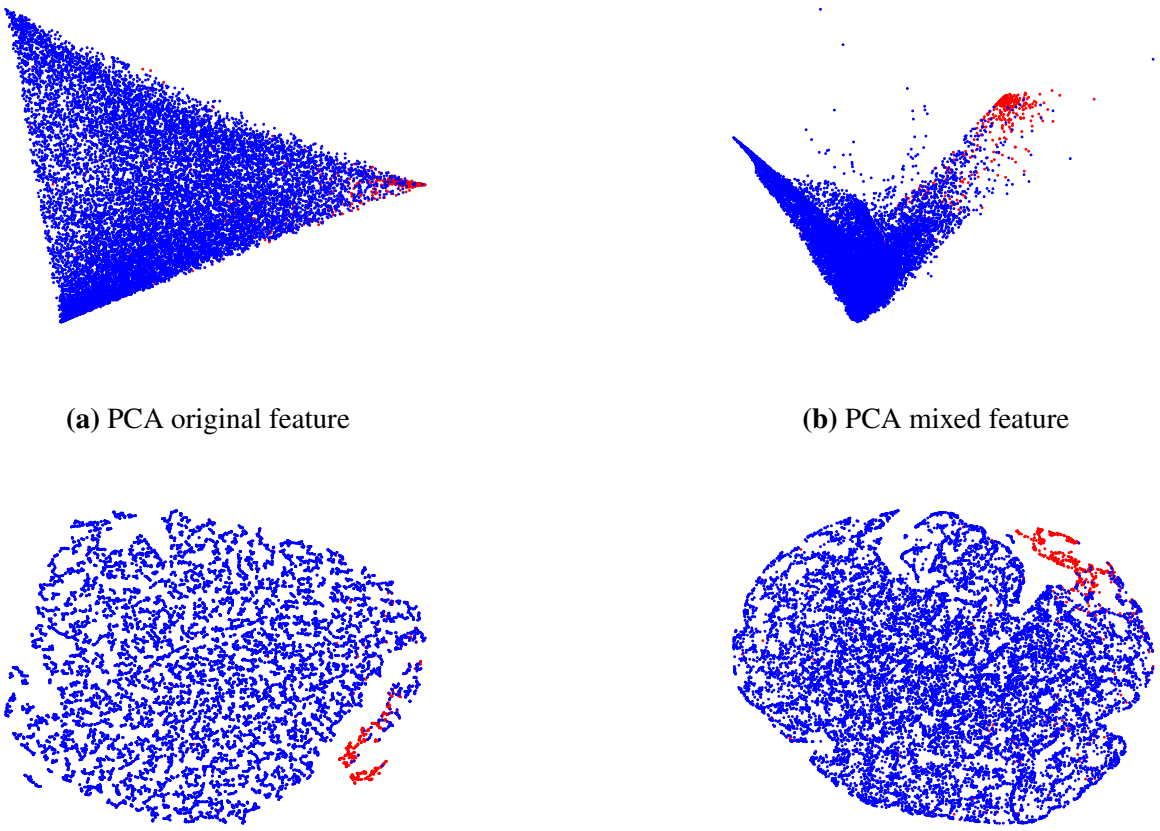
Figure 1. Aggregation pattern. Aggregate from typical and atypical perspectives. Solid lines represent the aggregation of the typical perspective, while dashed lines represent the aggregation of the atypical perspective. The thickness of the line is directly proportional to the weight value of a node.

Our main contributions are summarized as follows:

- We transform features to generate better prototypes, Those new features can also eliminate the unnecessary information and increase the separation between fraudulent nodes and benign ones. Results are visualized in Figure 2. Therefore, the GNN can more effectively identify fraudulent nodes within the graph.
- We utilize the GCD of each node to extract information on a global scale, which offers a novel perspective for observing fraud patterns, ensures model performance and significantly enhances convergence speed.
- We aggregate both typical and atypical information as shown in Figure 1. This approach enriches the message source and removes disruptive information, directly enhancing model performance.

In addition, extensive experiments are conducted on two open datasets. The outcome shows that our model outperforms the state-of-the-art model.

To accommodate different requirements, we provide two versions of our methods. The lightweight version delivers solid performance with fast processing, while the full version provides superior performance among baseline models with relatively fast speed.



(a) PCA original feature

(b) PCA mixed feature

(c) T-SNE original feature

(d) T-SNE mixed feature

Figure 2. Feature embeddings on T-Finance visualization using two different dimensionality reduction techniques. Red color represents fraudulent nodes, while blue represents benign nodes.

2. Related Work

2.1. Financial Fraud Detection

Several machine learning techniques have been proposed to address the problem of financial fraud detection. For example, reference [15] compare neural network-based models and decision tree models, finding that neural networks outperform decision trees. Additionally, a signature-based method for detecting potential fraud in e-commerce applications was proposed by reference [16]. This approach provides an alternative method for detecting fraudulent activities by identifying deviations in user behavior, thereby enabling real-time detection of potential fraudulent activities. Moreover, A deep learning-based model that integrates numerical financial data with textual information from management discussions [17] has been developed to enhance the detection of financial statement fraud among Chinese listed companies. This model demonstrates significant improvements over traditional methods. Futhermore, reference [18] presents a novel semi-supervised Group-based Fraud Detection Network (GFDN) that leverages structural, attribute, and community information from attributed bipartite graphs to effectively detect group-based financial fraud on e-commerce platforms.

2.2. Graph Anomaly Detection

Fraudulent activities have become increasingly frequent, leading to the development of various detection methods. Rule-based and outlier detection techniques, as summarized in [19,20], highlight models based on machine learning approaches, including support vector machines (SVM) and decision trees.

Recently, graph neural networks(GNN) have been utilized in fraud detection. For instance, Care-GNN [4] and Rio-GNN [21] exploit reinforcement learning to detect camouflage activities within networks. PCGNN [7] connects homophilic nodes and filters out heterophilic nodes to enhance the message passing process. Additionally, [22,9] utilize label information, dividing nodes into separate groups based on their labels and separately processing messages generated from different groups.

The prototype has been employed in previous studies [12,13] for feature optimization, enhancing the network's ability to distinguish between fraudulent and benign nodes. However, these methods incorporate the prototype only within the training loss, neglecting the critical confidence information that indicates whether a node in the graph is typical or atypical. This oversight restricts the potential benefits of using the prototype for more nuanced and effective differentiation.

3. Methodology

Previous models often encounter the issue of message elimination in resource-intensive methods like reinforcement learning or graph transformers. In contrast, some newest models avoid message elimination by dividing neighbors into distinct groups and aggregating their information separately. These operations also increase model complexity and extend training and inference time. However, by using GCD, our model achieves better performance, enabling faster training and inference simultaneously.

In this section, we outline the GCD-GNN framework. First, we define the role of GCD within the fraud detection context in Section 3.1. Then, an overview of the entire model is provided in Section 3.2. Finally, we detail the key components in Sections 3.3.– 3.6.

3.1. Prototype and Global Confidence Degree (GCD)

Definition 1 (Multi-relation Graph)**.** *We define a multi-relation graph as* $\mathcal{G} = (\mathcal{V}, \mathcal{X}, \{\mathcal{E}_r\}_{r=1}^{R}, \mathcal{Y})$. $\mathcal{V}$ *is the set of nodes* $\{v_1, \ldots, v_n\}$. *Each node* v_i *has a d-dimensional feature vector* $\mathbf{x}_i \in \mathbb{R}^d$ *and* $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ *is the features.* $e^r_{i,j} = (v_i, v_j) \in \mathcal{E}_r$ *is an edge between* v_i *and* v_j *with a relation* $r \in \{1, \ldots, R\}$. *Note that an edge can be associated with multiple relations and there are R different types of relations.* $\mathcal{Y} = \{\mathbf{y}_1, ..., \mathbf{y}_n\}$ *is the set of labels for each node in* $\mathcal{V}$.

In our scenario, $\mathcal{Y} \in \{fr, be, un\}$, where fr means fraud labeled nodes, be means benign labeled nodes and un means unlabeled nodes.

Definition 2 (Prototype)**.** *We define Prototype as* $\mu \in \mathbb{R}^d$. Φ*:* $\mathbb{R}^d \to \mathbb{R}^d$ *refers to the transformation applied to features.* ξ*:* $\mathbb{R}^{n \times d} \to \mathbb{R}^d$ *aggregates features into a single vector.*

$$\begin{aligned} \mu_{fr} &= \xi(concat(\Phi(\mathbf{x}_i))) \quad \mathbf{y}_i = fr, \\ \mu_{be} &= \xi(concat(\Phi(\mathbf{x}_j))) \quad \mathbf{y}_j = be. \end{aligned} \tag{1}$$

Further details of σ and ξ are provided in Section 3.3.

Definition 3 (Global Confidence Degree)**.** *we denote Global Confidence Degree (GCD) as g.* $g_i \in \mathbb{R}$ *is the GCD value of the i-th node in the graph.* $\sigma : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$*, means the similarity function that measures the difference of two features.*

$$g_i = \begin{cases} \sigma(\mu_{\mathbf{y}_i}, \mathbf{x}_i) & \text{if } \mathbf{y}_i = fr \text{ or } be, \\ \max(\sigma(\mu_{fr}, \mathbf{x}_i), \sigma(\mu_{be}, \mathbf{x}_i)) & \\ & \text{if } \mathbf{y}_i = un. \end{cases} \tag{2}$$

g_i represents the typicality of the node i. For labeled nodes, we use the similarity between each node and its corresponding prototype. For unlabeled nodes, we select the maximum between $\sigma(\mu_{fr}, x_i)$ and $\sigma(\mu_{be}, x_i)$. Details about the similarity function can be found in Section 3.5.

3.2. Overview

GCD-GNN includes a prototype calculator, a GCD estimator, a special GNN layer and a multilayer perceptron (MLP) aggregator. The prototype and GCD estimator contains an iterative prototype generator and a GCD generator depends on the similarity between nodes and their corresponding prototypes. The special GNN layer based on GraphSAGE [23], contains a message generator utilizing two kinds weight values generated by original GCD and its reverse. An aggregator receives messages derived from two kinds of weight values. The detailed structure of our method is shown in Figure 3.

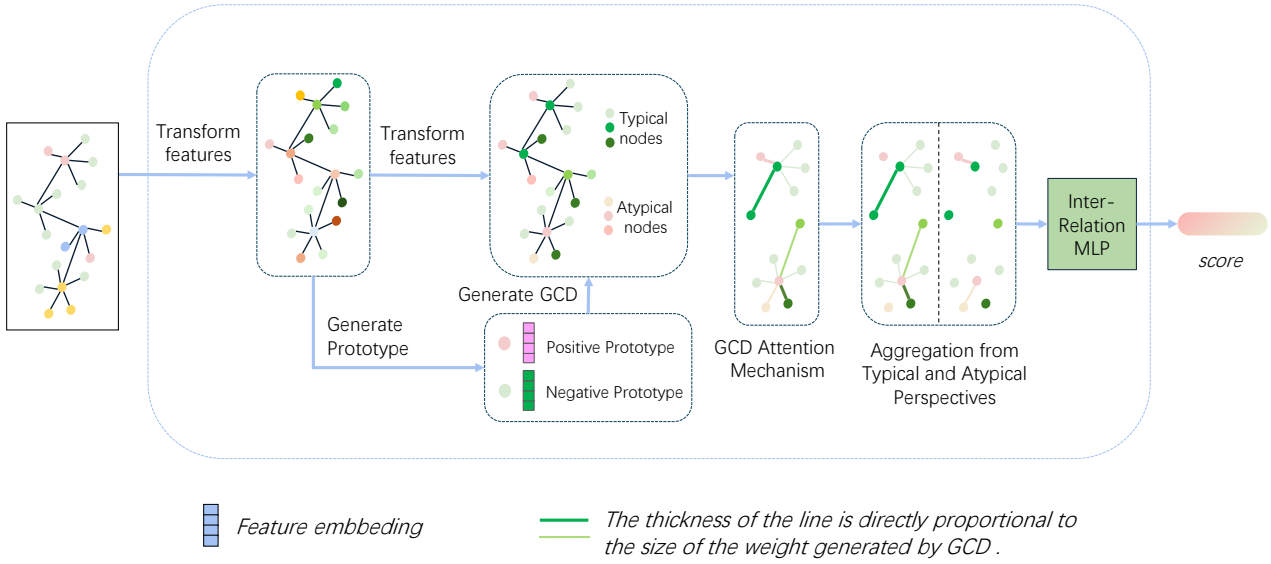


Figure 3. An illustration of the proposed framework.

3.3. Extracting Prototype Feature

Inspired by reference [12], to extract the prototype feature, we exploit the iterative extraction of the prototype. Firstly, we use an MLP and Graph Normalization [24] to process the initial features, projecting

those features into a space that is fitting for measuring similarity.

$$\mathbf{x}_{exp} = GraphNorm(\Phi(\mathscr{X})), \tag{3}$$

where Φ indicates an MLP. In addition, prototypes are generated by calculating the mean value of node features for the corresponding category. After this initial state, prototypes are iteratively updated based on node similarity, as shown in Equation1. Here, Φ represents an MLP. For the initial state, ξ employs the Mean function, which calculates the average value of a set of features. For subsequent updates, we adopt the strategy proposed by reference [12], with the following specifics:

$$\begin{aligned} s_v^{(e)} &= cos(\mathbf{x}_v^{(e)}, \mu^{(e-1)}), \\ w_v^{(e)} &= \frac{exp(s_v^{(e)}/\tau)}{\sum_{u=1}^{N} exp(s_u^{(e)/\tau})}, \\ \mu^{(e)} &= \sum_{v=1}^{N} w_v \cdot \mathbf{x}_v^{(e)}, \end{aligned} \tag{4}$$

where τ is the temperature parameter that controls the smoothness of the weights. First, we compute the cosine similarity between each node v and the previous prototype $\mu^{(e-1)}$, as shown in the first part of Equation 4. The softmax output of this similarity serves as the weight $w_v^{(e)}$ for each node. Finally, the updated prototype $\mu^{(e)}$ is calculated as a weighted sum of the node features.

3.4. Utilizing the Projected Node Feature

We propose a weight mix method to leverage the projected features while preserving the essential characteristics of the original features that might be lost during projection. The details of this method are as follows:

$$\begin{aligned} \lambda &= Sigmoid(\Phi(\mathbf{x})), \\ \mathbf{x}_{mixed} &= \lambda \cdot \mathbf{x}_{exp} + (1-\lambda) \cdot \mathbf{x}, \end{aligned} \tag{5}$$

where Φ indicates an MLP which generates a number that is processed by a *Sigmoid* function to range from 0 to 1 and this result determine the weight λ. Subsequently, we combine the two features by adding them according to the weight λ. We use PCA and T-SNE to visualize the effect of the mixed feature. The details are in Figure 2.

3.5. Global Confidence Degree Calculation

To calculate the GCD, we need to calculate similarity by comparing each node feature with the corresponding prototype. We calculate the similarity value using the cosine function as follows:

$$\sigma(\mu_c, \mathbf{x}_i) = \cos(\mu_c, \mathbf{x}_i). \tag{6}$$

The strategy for processing labeled and unlabeled data to calculate GCD is mentioned in Equation 2.

3.6. Aggregation from Typical and Atypical Perspectives

To utilize GCD, two perspectives, termed typical and atypical, are employed for message generation. In the typical perspective, GCD is unchanged from the original one defined in Def. 3. The atypical GCD is the inversion of the typical GCD, *i.e.* represented as the negative of the original GCD.

$$\begin{aligned} g_i^{typ} &= g_i, \\ g_i^{atyp} &= -g_i. \end{aligned} \tag{7}$$

When a node needs to aggregate messages, the GCD of its neighbors is used to generate the corresponding message weights. In order to make the weights generated by the GCD more effective, according to reference [25], we use a GCD attention mechanism similar to graph attention network.

$$\begin{aligned} w_{ij} &= LeckyRelu(g_j), \\ \alpha_{ij} &= \frac{exp(w_{ij})}{\sum_{k \in \mathcal{N}_i} exp(w_{ik})}, \end{aligned} \tag{8}$$

where i is a target node and j is one of its neighbors. $\mathcal{N}_i$ means the neighbor set of node i. α_{ij} means the final weight used in message aggregation. When we use g_i^{typ} in the Equation 8, we denote the weight as α_{ij}^{typ}. Similarly, g_i^{atyp} corresponds to α_{ij}^{atyp}.

To utilize local information, according to reference [9], a self-feature matrix is calculated by multiplying the node feature by the trained parameter. The message passing period is affected by the node feature.

$$\begin{aligned} W_i^{typ} &= \Psi_i^{typ}(\mathbf{x}_i), \\ W_i^{atyp} &= \Psi_i^{atyp}(\mathbf{x}_i), \end{aligned} \tag{9}$$

where $\Psi_i^{\text{typ}}(\mathbf{x}_i)$ and $\Psi_i^{\text{atyp}}(\mathbf{x}_i)$: $\mathbb{R}^d \to \mathbb{R}^{d \times d'}$ are two learnable weight generators. Each node receives an individual transformation weight matrix.

The message generation process, which utilizes both typical and atypical perspectives, is as follows:

$$m_i = W_i^{typ} \sum_{j \in \mathcal{N}_i} (\alpha_{ij}^{typ} \mathbf{x}_j) + W_i^{atyp} \sum_{j \in \mathcal{N}_i} (\alpha_{ij}^{atyp} \mathbf{x}_j). \tag{10}$$

3.7. Lightweight Model

The lightweight version of our method consists of prototype extracting, feature optimization and GCD attention machenism, mentioned in Section 3.3.–3.5., Building on this foundation, the full version adds self-feature matrix and aggregation from typical an atypical perspective, mentioned in 3.6. The lightweight model has fast training and inference speed and could achieve solid performance. The details of which are in Sections 4.2., 4.3.

4. Experiment

4.1. Experimental Setup

4.1.1. Datasets

- **T-Finance dataset** [26] aims to identify anomalous accounts in transaction networks. The nodes represent unique anonymized accounts, each characterized by 10-dimensional features related to registration days, logging activities, and interaction frequency. The edges in the graph denote transaction records between accounts. Human experts annotate nodes as anomalies if they fall into categories such as fraud, money laundering, or online gambling.
- **FDCompCN dataset** [27] detect financial statement fraud in Chinese companies. This dataset constructs a multi-relation graph based on supplier, customer, shareholder, and financial information from the China Stock Market and Accounting Research (CSMAR) database. It includes data from 5,317 publicly listed companies on the Shanghai, Shenzhen, and Beijing Stock Exchanges between 2020 and 2023. FDCompCN features three relations: C-I-C (investment relationships), C-P-C (companies and their disclosed customers), and C-S-C (companies and their disclosed suppliers).

Detailed statistics for the two datasets are presented in Appendix.

4.1.2. Comparison Methods

We compare our method with two types of models. (1) general models, including GCN [28], GAT [25], and GraphSAGE [23]; and (2) those specifically optimized for fraud detection using GNNs, including Care-GNN [4], PC-GNN [7], BWGNN [26], Split-GNN [27], GHRN [29], and PMP [9]. For detailed descriptions of these baselines, please refer to Appendix.

According to reference [26], we adopt data splitting ratios of 40%:20%:40% for the training, validation, and test sets in the supervised scenario. To ensure consistency in our evaluations, each model underwent 5 trials with different random seeds. We present the average performance and standard deviation for each model as benchmarks for comparison. For clarity in the paper, all average values in the tables have been scaled by a factor of 100, and standard deviations by a factor of 10.

4.2. Performance Comparasion

The details of our model are introduced in Section 3. Two kinds of GCD-GNN are provided. The lightweight model, GCD-GNN$_{light}$, contains feature optimization and GCD attention mechanisms. The

full model, GCD-GNN, which includes all components, additionally integrates self-feature matrix and aggregation from typical and atypical perspectives on the basis of the lightweight model.

Table 1. Experiment results on T-Finance and FDCompCN.

	T-Finance			FDCompCN		
Method	AUC	F1-Macro	G-Mean	AUC	F1-Macro	G-Mean
GCN	$92.76_{\pm 0.13}$	$65.63_{\pm 1.15}$	$84.28_{\pm 0.27}$	$59.60_{\pm 0.27}$	$45.84_{\pm 0.49}$	$56.67_{\pm 0.24}$
GAT	$93.04_{\pm 0.28}$	$77.70_{\pm 0.50}$	$83.52_{\pm 1.00}$	$59.08_{\pm 0.19}$	$45.97_{\pm 0.47}$	$52.66_{\pm 0.30}$
GraphSAGE	$84.02_{\pm 0.33}$	$70.56_{\pm 0.90}$	$79.67_{\pm 0.53}$	$63.31_{\pm 0.09}$	$45.97_{\pm 0.26}$	$52.66_{\pm 0.30}$
Care-GNN	$87.22_{\pm 0.51}$	$74.42_{\pm 0.72}$	$60.71_{\pm 1.31}$	$57.36_{\pm 0.05}$	$47.79_{\pm 0.15}$	$50.96_{\pm 0.39}$
PC-GNN	$93.49_{\pm 0.07}$	$81.57_{\pm 0.38}$	$80.97_{\pm 0.73}$	$59.76_{\pm 0.58}$	$23.83_{\pm 0.92}$	$54.69_{\pm 0.53}$
BWGNN	$93.68_{\pm 0.15}$	$84.15_{\pm 0.31}$	$78.79_{\pm 0.51}$	$61.59_{\pm 0.62}$	$44.88_{\pm 1.18}$	$54.69_{\pm 0.53}$
Split-GNN	$95.51_{\pm 0.07}$	$82.29_{\pm 0.05}$	$84.47_{\pm 0.25}$	$62.85_{\pm 0.39}$	$45.40_{\pm 0.57}$	$55.56_{\pm 0.70}$
GHRN	$95.78_{\pm 0.08}$	$89.01_{\pm 0.03}$	$84.86_{\pm 0.11}$	$62.09_{\pm 0.57}$	$47.45_{\pm 0.85}$	$54.60_{\pm 0.48}$
PMP	$97.07_{\pm 0.01}$	$91.96_{\pm 0.04}$	$88.53_{\pm 0.09}$	$54.34_{\pm 0.06}$	$48.38_{\pm 0.14}$	$12.02_{\pm 1.05}$
GCD-GNN$_{light}$ (Ours)	$97.06_{\pm 0.01}$	$92.13_{\pm 0.01}$	$88.45_{\pm 0.07}$	$71.01_{\pm 0.12}$	$58.12_{\pm 0.15}$	$\mathbf{62.51}_{\pm 0.31}$
GCD-GNN (Ours)	$\mathbf{97.26}_{\pm 0.01}$	$\mathbf{92.37}_{\pm 0.05}$	$\mathbf{88.62}_{\pm 0.11}$	$\mathbf{71.72}_{\pm 0.18}$	$\mathbf{59.68}_{\pm 0.31}$	$57.99_{\pm 0.31}$

Table 2. Ablation results on T-Finance and FDCompCN.

	T-Finance			FDCompCN		
Method	AUC	F1-Macro	G-Mean	AUC	F1-Macro	G-Mean
GraphSAGE	$84.02_{\pm 0.33}$	$70.56_{\pm 0.90}$	$79.67_{\pm 0.53}$	$63.31_{\pm 0.09}$	$45.97_{\pm 0.26}$	$52.66_{\pm 0.30}$
M1	$97.06_{\pm 0.01}$	$92.13_{\pm 0.01}$	$88.45_{\pm 0.07}$	$71.01_{\pm 0.12}$	$58.12_{\pm 0.15}$	$\mathbf{62.51}_{\pm 0.31}$
M2	$97.14_{\pm 0.01}$	$92.07_{\pm 0.03}$	$88.19_{\pm 0.10}$	$70.58_{\pm 0.28}$	$58.86_{\pm 0.26}$	$58.48_{\pm 0.44}$
M3	$\mathbf{97.26}_{\pm 0.01}$	$\mathbf{92.37}_{\pm 0.05}$	$\mathbf{88.62}_{\pm 0.11}$	$\mathbf{71.72}_{\pm 0.09}$	$\mathbf{59.68}_{\pm 0.28}$	$57.99_{\pm 0.22}$

The results are reported in Table 1, which demonstrate that our light version model performs better than baseline models on most metrics in the public datasets. Furthermore, our complete model comprehensively surpasses the lightweight model and outperforms the baseline models across all metrics.

We also compared the convergence speed of all models on the T-Finance dataset. The results are presented in Figure 4, with detailed values provided in the Appendix. The results indicate that our lightweight model achieves a high AUC level in a short period of time. Furthermore, the full model achieves the highest score within a medium timeframe. The outstanding performance of our model arises from the fact that generic GNNs fail to consider the importance of each sample and aggregate messages uniformly. In contrast, our model leverages the GCD to evaluate whether the information from neighboring nodes is typical or not, which significantly improves the performance and boosts the training speed, thereby reducing computational resource consumption.

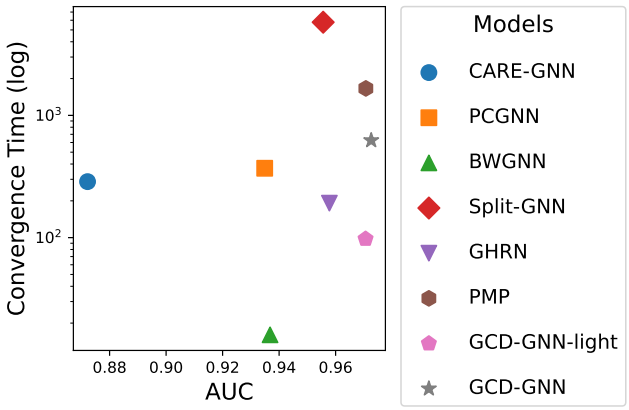


Figure 4. Convergence time (log) and AUC of models on T-Finance.

4.3. Ablation Study

We conduct an ablation study to verify the impact of each component, using GraphSAGE as the benchmark model. Three components evaluated are as follows:

- M1 indicates prototype extracting, feature optimization and GCD attention machenism, mentioned in Sections 3.3.–3.5.
- M2 indicates the self-feature matrix, mentioned in Equation 9.
- M3 indicates aggregation from typical and atypical perspectives, mentioned in Section 3.6.

The results are provided in Table 2, indicating that GraphSAGE demonstrates poor performance across all metrics, highlighting its limitations in identifying financial fraud patterns. Conversely, our model exhibits significant improvements in all metrics after incorporating feature transformation and GCD attention mechanisms, which are central to our approach. This underscores the pivotal role of GCD in financial fraud detection. The inclusion of M2 and M3 further enhances the performance of our model, elevating it to a higher level.

4.4. Impact of GCD on Model Message Aggregation

To explore the impact of GCD on model performance and analyze the relationships between nodes and their neighbors from both typical and atypical perspectives. For typical perspective, we examine the typical GCD attentive Euclidean distances $d_i^{typ} = \frac{\sum_{j\in\mathcal{N}_i} \alpha_{ij}^{typ} \|\mathbf{x}_j - \mathbf{x}_i\|}{\sum_{j\in\mathcal{N}_i} \alpha_{ij}^{typ}}$, where α_{ij}^{typ} is calculated as the method in Section 3.6. For comparision, we also calculate the average Euclidean distances. We randomly choose 20 nodes with neighbors on T-Finance and FDCompCN datasets. The rate of change represents the ratio of the typical GCD-weighted distance to the original distance. The results are reported in Figure 5.

The results show that the rate of change is inversely proportional to the node's GCD. This indicates that higher GCD node tend to aggregate information from closer nodes, while lower GCD tend to aggregate information from more distant nodes. This strategy suggests that nodes with high GCD, which are more typical or representative, tend to aggregate less diverse information, as their characteristics already strongly indicate their belonging to a certain category. Conversely, nodes with lower GCD lack direct distinguishing features and thus tend to rely on diverse information from distant nodes. This strategy

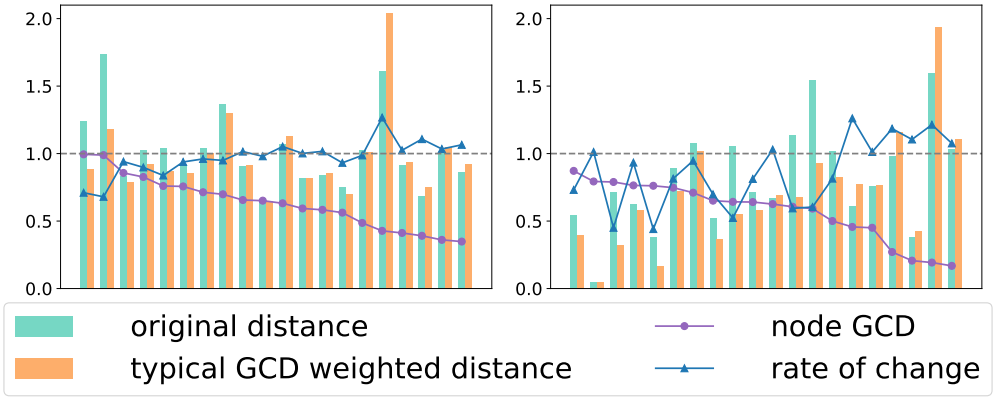


Figure 5. Distances analyze on T-Finance (left) and FDCompCN (right).

also ensures that the aggregated information predominantly comes from nodes with higher GCD, making the aggregated information more reliable.

For atypical perspective, the presence of g^{atyp} allows for the capture of outlier information. We calculate atypical GCD attentive Euclidean distances $d_i^{atyp} = \frac{\sum_{j\in\mathcal{N}_i}\alpha_{ij}^{atyp}\|\mathbf{x}_j-\mathbf{x}_i\|}{\sum_{j\in\mathcal{N}_i}\alpha_{ij}^{atyp}}$, where α_{ij}^{atyp} is calculated as the method in Section 3.6. We find that, d_i^{atyp} tends to be larger compared to d_i^{typ}, indicating that extra diverse information can be aggregated from the atypical perspective to aid classification. Detailed results are provided in Appendix.

We analyze the F1-Macro value in the different range of GCD on T-Finance and FDCompCN datasets. The result are reported in Figure 6 and more metrics analysis is in Appendix. We compare our model with the most competitive model PMP [9]. As shown in Figure 6. We find that GCD-GNN outperforms in most range of GCD, from low to high concretely from 0.1 to 0.8, which demonstrates that: (1) nodes with low GCD absorb more information that differs from their own features, (2) nodes with high GCD absorb more similar features, and (3) incorporating atypical information positively impacts model performance.

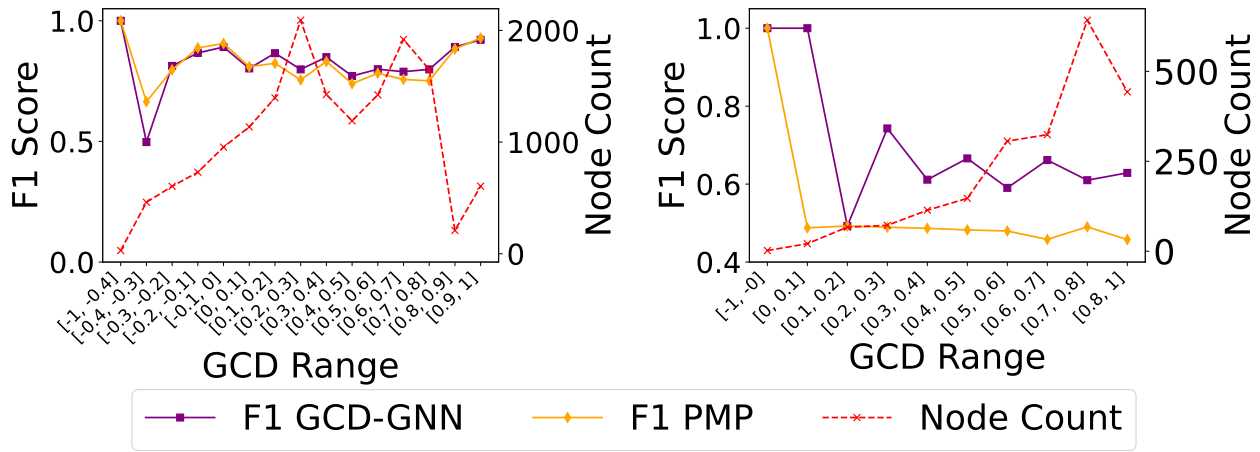


Figure 6. F1-Macro score on GCD-GNN and PMP across different ranges of GCD on T-Finance (left) and FDCompCN (right).

4.5. *Sensitive Analyze*

We explore the model's sensitivity to the important parameters GCD drop rate and hidden dimenssion. All results are presented in Figure 7, where GCDR means GCD drop rate and HD means hidden dimension. Detailed values in the figure are provided in the Appendix.

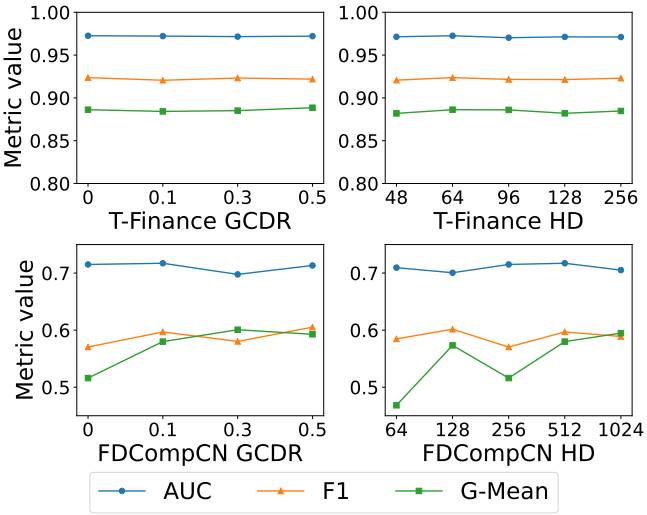


Figure 7. Hyperparameters sensitive results.

- **The GCD Drop Rate.** During the training of our model, we observed potential overfitting when generating weights through GCD attention mechanism. To address this, in addition to the dropout layer at the end of the network, a GCD dropout layer was incorporated into the model. As a result, we find that the optimal GCD drop rate with the highest AUC is 0 for T-Finance and 0.1 for FDCompCN, suggesting that T-Finance avoids overfitting during the generation of weights, whereas FDCompCN may suffer from slight overfitting.
- **The Hidden Dimenssion.** The hidden dimension in the model is also crucial to performance; A low hidden dimension leads to inadequate explanation of data features, while a high hidden dimension can result in overfitting. We find that the model performs best with a feature dimension of 48 on T-Finance, while FDCompCN achieves optimal performance with a dimension of 512, which is proportional to the feature dimension of the respective datasets.

5. Conclusion

In this work, we introduce the concept of GCD and define its role in the process of information aggregation. We analyze the reasoning behind the effectiveness of GCD in enhancing the detection of fraudulent activities and propose a novel GNN-based model named GCD-GNN. Specifically, our model utilizes GCD for feature optimization, message filtering and aggregation from typical and atypical perspectives. Experimental results demonstrate that GCD-GNN outperforms state-of-the-art methods in terms of AUC, F1-Macro, G-Mean, and convergence speed. We also design a lightweight GCD-GNN (GCD-GNN$_{light}$) that outperforms the baselines on almost all metrics, is slightly weaker than GCD-GNN on fraud detection, but obviously outperforms it in convergence and inference speed.

Authors' contribution

Conceptualization, Jiaxun Liu, Yue Tian and Guanjun Liu; methodology, Jiaxun Liu and Yue Tian; software, Jiaxun Liu; validation, Yue Tian and Guanjun Liu.; formal analysis, Jiaxun Liu; investigation, Jiaxun Liu and Yue Tian; resources, Guanjun Liu; data curation, Jiaxun Liu; writing—original draft preparation, Jiaxun Liu and Yue Tian; writing—review and editing, Yue Tian, Guanjun Liu; visualization, Jiaxun Liu and Yue Tian; supervision, Guanjun Liu; project administration, Guanjun Liu; funding acquisition, Guanjun Liu. All authors have read and agreed to the published version of the manuscript.

Conflicts of interests

The authors declare no conflict of interest.

Appendix A: implementation details

The proposed GCD-GNN provides an implementation in PyTorch. All experiments are run on a server with 32 cores, 120 GB memory, 1 NVIDIA RTX 4090 GPU, and Ubuntu 20.04 as the operating system. The hyper-parameter setting of GCD-GNN is listed in Table A3.

Table A3. Hyper-parameters setting on T-Finance and FDCompCN datasets.

Parameter	T-Finance	FDCompCN
learning rate	0.005	0.005
batch size	1024	128
dropout	0.292	0
hidden dimension	64	512
n layer	1	1
weight decay	0	0
optimizer	Adam	Adam
thres	0.5	0.5
GCD drop	0	0.1

We use grid search to find the best hyperparameters, with results rounded to three decimal places. Detailed results can be found in the configuration files in the config directory within the code. The code is publicly available on Github..

Appendix B: metrics

Following reference [26], we use AUC, F1-Macro and G-Mean as our experiments metrics. AUC measures the area under the ROC curve and reflects the model's ability to distinguish between positive and negative classes across all possible classification thresholds. F1-Macro calculates the F1 score for each class

https://github.com/GCDGNN/GCD-GNN/

independently and then takes the average. The G-Mean, or geometric mean, is the square root of the product of sensitivity and specificity, showing the balance between true positive rate and true negative rate. Higher values for these metrics indicate better method performance.

Appendix C: baseline models introduction

In this section, we describe the baseline models used for comparison.

The general models are as follows:

- GCN [28], A graph convolutional network utilizing the first-order approximation of localized spectral filters on graphs.
- GAT [25], A graph attention network that employs the attention mechanism for neighbor aggregation.
- GraphSAGE [23], A graph neural network model based on sampling a fixed number of neighbor nodes.

The fraud detection models are as follows:

- Care-GNN [4], A camouflage-resistant GNN that enhances the aggregation process with three unique modules designed to counter camouflages and incorporates reinforcement learning.
- PC-GNN [7], A GNN-based method for addressing category imbalance in graph-based fraud detection through resampling techniques.
- BWGNN [26], A graph neural network utilizing a label-aware high-frequency indicator to prune the heterogeneous edges, effectively reducing heterophily and boosting graph anomaly detection performance.
- SplitGNN [27], A spectral GNN that addresses fraud detection in heterophilic graphs by splitting the graph into subgraphs and applying band-pass filters to capture diverse frequency signals.
- GHRN [29], A graph neural network using Beta wavelet filters to improve anomaly detection by addressing spectral energy 'right-shift' in large-scale datasets.
- PMP [9], A graph neural network enhancing fraud detection by distinguishing between homophilic and heterophilic neighbors in message passing, addressing label imbalance and mixed homophily-heterophily.

Appendix D: training AUC and time details

In Table D4 we present the detailed AUC value and convergence time consumption.

Table D4. Traing AUC and time.

model	AUC	Time (s)
PCGNN	93.49	369.40
Care-GNN	87.22	287.09
BWGNN	92.33	16.04
SplitGNN	95.51	5592.14
GHRN	95.78	191.42
PMP	97.07	1661.78
GCD-GNN$_{light}$	97.06	97.68
GCD-GNN	97.26	624.38

Appendix E: sensitive analyze details

In Tables E5–E8, We present the detailed value of AUC, F1, G-Mean influenced by hyperparameters.

Table E5. Performance metrics for different hidden dimension on T-Finance.

hiddim	AUC	F1-Macro	G-Mean
48	$97.14_{\pm 0.01}$	$92.07_{\pm 0.03}$	$88.19_{\pm 0.10}$
64	$97.26_{\pm 0.01}$	$92.37_{\pm 0.05}$	$88.62_{\pm 0.11}$
96	$97.03_{\pm 0.02}$	$92.16_{\pm 0.02}$	$88.60_{\pm 0.07}$
128	$97.13_{\pm 0.01}$	$92.14_{\pm 0.03}$	$88.20_{\pm 0.12}$
256	$97.11_{\pm 0.02}$	$92.30_{\pm 0.01}$	$88.47_{\pm 0.06}$

Table E6. Performance metrics for FDCompCN with different hidden dimensions.

hiddim	AUC	F1-Macro	G-Mean
64	$70.95_{\pm 0.14}$	$58.46_{\pm 0.61}$	$46.83_{\pm 2.25}$
128	$70.06_{\pm 0.12}$	$60.16_{\pm 0.21}$	$57.33_{\pm 0.40}$
256	$71.51_{\pm 0.12}$	$57.04_{\pm 0.53}$	$51.60_{\pm 2.51}$
512	$71.72_{\pm 0.18}$	$59.68_{\pm 0.31}$	$57.99_{\pm 0.31}$
1024	$70.52_{\pm 0.10}$	$58.89_{\pm 0.18}$	$59.48_{\pm 0.20}$

Table E7. Performance metrics on T-Finance with different attention drop rates.

GCD_drop	AUC	F1-Macro	G-Mean
0	$97.26_{\pm 0.01}$	$92.37_{\pm 0.05}$	$88.62_{\pm 0.11}$
0.1	$97.22_{\pm 0.02}$	$92.06_{\pm 0.02}$	$88.42_{\pm 0.08}$
0.3	$97.16_{\pm 0.02}$	$92.32_{\pm 0.02}$	$88.51_{\pm 0.07}$
0.5	$97.21_{\pm 0.02}$	$92.19_{\pm 0.02}$	$88.85_{\pm 0.15}$

Table E8. Performance metrics for FDCompCN with different attention drop rates.

GCD_drop	AUC	F1-Macro	G-Mean
0	$71.51_{\pm 0.12}$	$57.04_{\pm 0.53}$	$51.60_{\pm 2.51}$
0.1	$71.72_{\pm 0.18}$	$59.68_{\pm 0.31}$	$57.99_{\pm 0.31}$
0.3	$69.77_{\pm 0.20}$	$58.02_{\pm 0.43}$	$60.07_{\pm 0.25}$
0.5	$71.33_{\pm 0.04}$	$60.53_{\pm 0.06}$	$59.27_{\pm 0.35}$

Appendix F: performance in the different range of GCD on T-Finance and FDCompCN datasets

We visualize AUC and F1-MARCO in diffrent range on the test set on T-Finance and FDCompCN datasets, as shown in Figure F8. The missing AUC values are due to the presence of only one category of nodes within the specific GCD range.

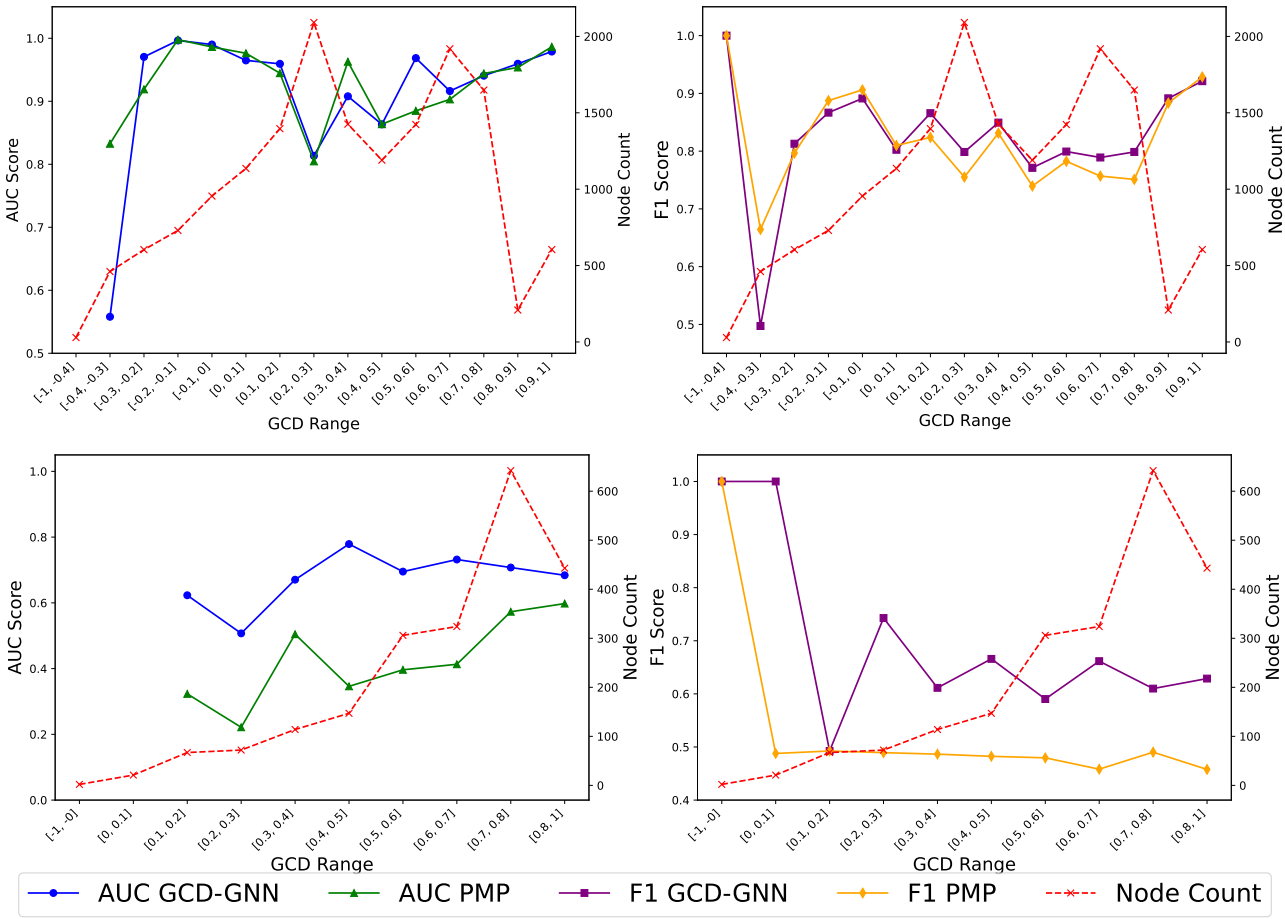


Figure F8. Performance in the different range of GCD on T-Finance (top) and FDCompCN (bottom) datasets

Appendix G: typical and atypical GCD weighted distance analysis

We calculate the atypical GCD weighted distance according to Section 4.4. Typical and atypical GCD weighted distances are calculated as follows:

$$d_i^{typ} = \frac{\sum_{j \in \mathcal{N}_i} \alpha_{ij}^{typ} \|\mathbf{x}_j - \mathbf{x}_i\|}{\sum_{j \in \mathcal{N}_i} \alpha_{ij}^{typ}},$$
$$d_i^{atyp} = \frac{\sum_{j \in \mathcal{N}_i} \alpha_{ij}^{atyp} \|\mathbf{x}_j - \mathbf{x}_i\|}{\sum_{j \in \mathcal{N}_i} \alpha_{ij}^{atyp}}, \tag{11}$$

where α_{ij}^{typ} and α_{ij}^{atyp} are calculated as the method mentioned in Section 3.6.

As the result shown in Figure G9, we find that, d_i^{atyp} tends to be larger compared to d_i^{typ}, indicating that extra diverse information can be aggregated from the atypical perspective to aid classification.

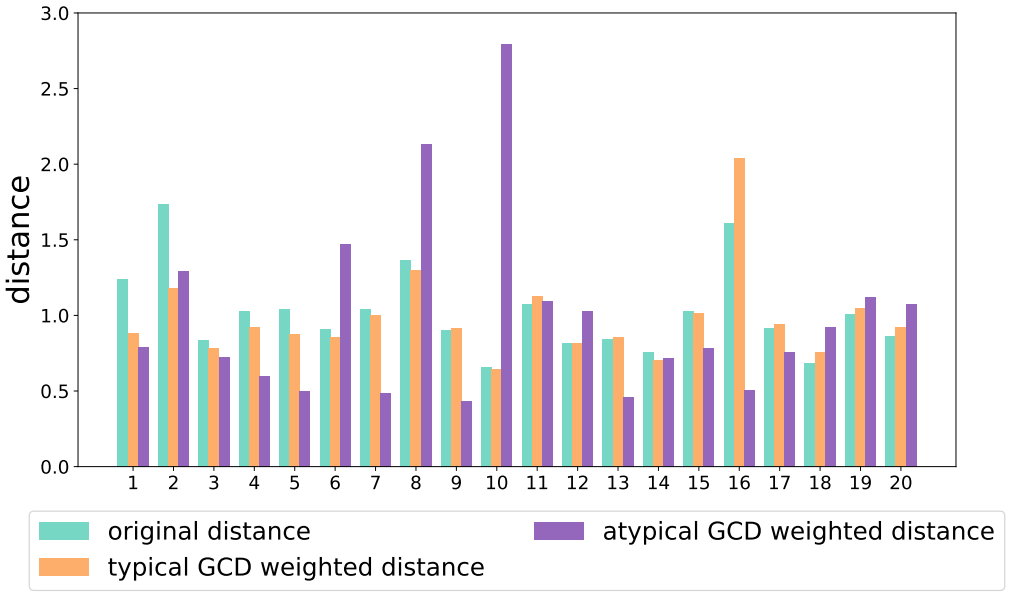


Figure G9. GCD weighted distance analysis

References

[1] Reurink A. Financial fraud: a literature review. *J. Econ. Surv.* 2018, 32(5):1292–1325.

[2] Weng H, Li Z, Ji S, Chu C, Lu H, *et al.* Online e-commerce fraud: a large-scale detection and analysis. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, Paris, France, April 16–19, 2018, pp. 1435–1440.

[3] Ma X, Wu J, Xue S, Yang J, Zhou C, *et al.* A comprehensive survey on graph anomaly detection with deep learning. *IEEE Trans. Knowl. Data Eng.* 2021, 35(12):12012–12038.

[4] Dou Y, Liu Z, Sun L, Deng Y, Peng H, *et al.* Enhancing graph neural network-based fraud detectors against camouflaged fraudsters. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, Virtual Event, Ireland, October 19–23, 2020, pp. 315–324.

[5] Wang D, Lin J, Cui P, Jia Q, Wang Z, *et al.* A semi-supervised graph attentive network for financial fraud detection. In *2019 IEEE international conference on data mining (ICDM)*, Beijing, China, November 08–11, 2019, pp. 598–607.

[6] Liu C, Sun L, Ao X, Feng J, He Q, *et al.* Intention-aware heterogeneous graph attention networks

for fraud transactions detection. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, Virtual Event, Singapore, August 14–18, 2021, pp. 3280–3288.

[7] Liu Y, Ao X, Qin Z, Chi J, Feng J, *et al.* Pick and choose: a GNN-based imbalanced learning approach for fraud detection. In *Proceedings of the web conference 2021*, Ljubljana, Slovenia, April 19–23, 2021, pp. 3168–3177.

[8] Wang Y, Zhang J, Huang Z, Li W, Feng S, *et al.* Label information enhanced fraud detection against low homophily in graphs. In *Proceedings of the ACM Web Conference 2023*, Austin, USA, April 30 - May 4, 2023, pp. 406–416.

[9] Zhuo W, Liu Z, Hooi B, He B, Tan G, *et al.* Partitioning message passing for graph fraud detection. *arXiv* 2024, arXiv:2412.00020.

[10] Roy A, Shu J, Li J, Yang C, Elshocht O, *et al.* GAD-NR: Graph anomaly detection via neighborhood reconstruction. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, Merida, Mexico, February 5–9, 2024, pp. 576–585.

[11] Ding K, Li J, Bhanushali R, Liu H. Deep anomaly detection on attributed networks. In *Proceedings of the 2019 SIAM international conference on data mining*, Anaheim, USA, May 2–4, 2019, pp. 594–602.

[12] Gao Y, Wang X, He X, Liu Z, Feng H, *et al.* Alleviating structural distribution shift in graph anomaly detection. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, Singapore, Singapore, February 27 - March 3, 2023, pp. 357–365.

[13] Shi F, Cao Y, Shang Y, Zhou Y, Zhou C, *et al.* H2-FDetector: a GNN-based fraud detector with homophilic and heterophilic connections. In *Proceedings of the ACM Web Conference 2022*, Virtual Event, Lyon France, April 25–29, 2022, pp. 1486–1494.

[14] Chen N, Liu Z, Hooi B, He B, Fathony R, *et al.* Consistency training with learnable data augmentation for graph anomaly detection with limited supervision. In *The Twelfth International Conference on Learning Representations*. Vienna, Austria, May 7–11, 2024. .

[15] Zaki MJ, Meira W Jr. *Data Mining and Analysis: Fundamental Concepts and Algorithms*. 2nd ed. Cambridge: Cambridge University Press, 2014.

[16] Mota G, Fernandes J, Belo O. Usage signatures analysis an alternative method for preventing fraud in E-Commerce applications. In *2014 International Conference on Data Science and Advanced Analytics (DSAA)*, Shanghai, China, October 30 - November 01, 2014, pp. 203–208.

[17] Wu XG, Du SY. An analysis on financial statement fraud detection for Chinese listed companies using deep learning. *IEEE Access* 2022, 10:22516–22532.

[18] Yu J, Wang H, Wang X, Li Z, Qin L, *et al.* Group-based Fraud Detection Network on e-Commerce Platforms. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, Long Beach, USA, August 6–10, 2023, pp. 5463–5475.

[19] Kou Y, Lu CT, Sirwongwattana S, Huang YP. Survey of fraud detection techniques. In *IEEE international conference on networking, sensing and control, 2004*, Taipei, China, March 21–23, 2004, pp. 749–754.

[20] Phua C, Lee V, Smith K, Gayler R. A comprehensive survey of data mining-based fraud detection research. *arXiv* 2010, arXiv:1009.6119.

[21] Peng H, Zhang R, Dou Y, Yang R, Zhang J, *et al.* Reinforced neighborhood selection guided multi-relational graph neural etworks. *ACM Trans. Inf. Syst.* 2021, 40(4):1–46.

[22] Wang Y, Zhang J, Huang Z, Li W, Feng S, *et al.* Label Information Enhanced Fraud Detection against Low Homophily in Graphs. In *Proceedings of the ACM Web Conference 2023*, Austin TX USAApril 30 - May 4, 2023, pp. 406–416.

[23] Hamilton W, Ying Z, Leskovec J. Inductive representation learning on large graphs. *Adv. Neural Inf. Process. Syst.* 2017, 30.

[24] Cai T, Luo S, Xu K, He D, Liu Ty, *et al.* Graphnorm: a principled approach to accelerating graph neural network training. In *International Conference on Machine Learning*, Virtual Conference, Canada, July 18–24, 2021, pp. 1204–1215.

[25] Veličković P, Cucurull G, Casanova A, Romero A, Lio P, *et al.* Graph attention networks. *arXiv* 2017, arXiv:1710.10903.

[26] Tang J, Li J, Gao Z, Li J. Rethinking graph neural networks for anomaly detection. In *International Conference on Machine Learning*, New Orleans, USA, July 17–23, 2022, pp. 21076–21089.

[27] Wu B, Yao X, Zhang B, Chao KM, Li Y. SplitGNN: Spectral Graph Neural Network for Fraud Detection against Heterophily. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, Birmingham, United Kingdom, 2023, pp. 2737–2746.

[28] Kipf TN, Welling M. Semi-supervised classification with graph convolutional networks. *arXiv* 2016, arXiv:1609.02907.

[29] Gao Y, Wang X, He X, Liu Z, Feng H, *et al.* Addressing heterophily in graph anomaly detection: A perspective of graph spectrum. In *Proceedings of the ACM Web Conference 2023*, Austin, USA, 2023, pp. 1528–1538.