Machine Learning (2024) 113:1069–1090
https://doi.org/10.1007/s10994-023-06464-0

Improving fraud detection via imbalanced graph structure learning

Lingfei Ren[1,2] · **Ruimin Hu**[1,2,5] · **Yang Liu**[3] · **Dengshi Li**[1,4] · **Junhang Wu**[1,2] · **Yilong Zang**[1,2] · **Wenyi Hu**[1,2]

Received: 11 January 2023 / Revised: 21 July 2023 / Accepted: 17 October 2023 /
Published online: 29 November 2023

Abstract

Graph-based fraud detection methods have recently attracted much attention due to the rich relational information of graph-structured data, which may facilitate the detection of fraudsters. However, the GNN-based algorithms may exhibit unsatisfactory performance faced with graph heterophily as the fraudsters usually disguise themselves by deliberately making extensive connections to normal users. In addition to this, the class imbalance problem also causes GNNs to overfit normal users and perform poorly for fraudsters. To address these problems, we propose an Imbalanced Graph Structure Learning framework for fraud detection (IGSL for short). Specifically, nodes are picked with a devised multi-relational class-balanced sampler for mini-batch training. Then, an iterative graph structure learning module is proposed to iteratively construct a global homophilic adjacency matrix in the embedding domain. Further, an anchor node message passing mechanism is proposed to reduce the computational complexity of the constructing homophily adjacency matrix. Extensive experiments on benchmark datasets show that IGSL achieves significantly better performance even when the graph is heavily heterophilic and imbalanced.

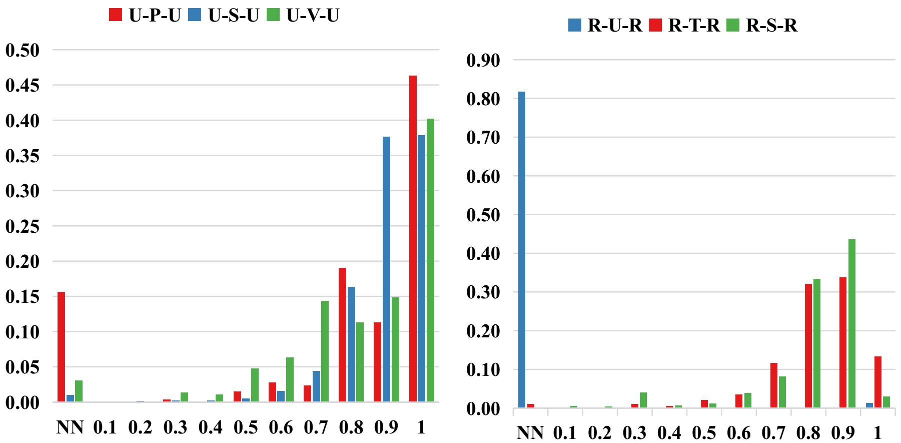
Keywords Fraud detection · Graph structure learning · Homophily · Heterophily

1 Introduction

With the booming of the Internet and telecommunication industries, various fraud activities have emerged in the fields of finance (Wang et al., 2019; Zhong et al., 2020), social security (Van Vlasselaer et al., 2017), and healthcare (Zhang et al., 2022), leading to user privacy breaches, personal property losses, and so on. Since frauds often occur in graph-like data such as the Internet, graph-based methods are widely used to detect fraudsters. Compared with traditional graph-based methods, GNN-based methods leverage the rich feature information and structural information of fraudsters, and thus gain more and more attention.

Editors: Dino Ienco, Roberto Interdonato, Pascal Poncelet.

Extended author information available on the last page of the article



(a) Amazon (b) Yelpchi

Fig. 1 Evidence of graph heterophily. The x-coordinate represents the graph heterophily ratios and y-coordinate represents the proportion of fraud nodes with corresponding graph heterophily ratios." NN" denotes a fraud node with no neighbors

Although these GNN-based fraud detection methods have made much progress, there still exist the following two main challenges.

Graph Heterophily. Generally, GNNs are essentially established on the homophily assumption that linked nodes tend to share similar features or have the same labels(Hamilton et al., 2017; Abu-El-Haija et al., 2019), which is exactly the opposite of fraud detection (Liu et al., 2021). Specifically, as confirmed by some researchers(Kaghazgaran et al., 2019; Ge et al., 2018), fraudsters deliberately make extensive connections with normal users and try to prevent contact with fraudsters to camouflage themselves, leading to an increase in graph heterophily in the local social network. To illustrate this phenomenon with statistics, we calculate the heterophily ratio of heterophilic edges to all adjacent edges of each fraudster in different relational subgraphs (i.e., meta-paths) from two opinion fraud detection datasets Amazon and Yelpchi (see Section 5.1 for details), and calculate the proportion of the number of fraudsters with the corresponding heterophily ratio to all the fraudsters in the whole graph from low to high. As Fig. 1 shows, these relational subgraphs, except the relation R-U-R of YelpChi, include a large number of fraudsters whose heterophily ratio is very high. Specifically, more than 80% of the fraudsters on Amazon have more than 50% heterophily; more than 70% of the fraudsters on the relation R-T-R and the relation R-S-R of Yelchi have more than 50% heterophily ratio. It is worth noting that more than 35% of fraudsters on Amazon have 100% heterophilic edges, which means that when simply aggregating their neighbor attributes based on GNNs, the fraudster's features will be completely swamped inside the normal users and will be difficult to identify correctly. To address this problem, several improved GNN-based algorithms have been proposed and they mainly fall into three directions, namely neighbor filtering-based methods (Dou et al., 2020; Liu et al., 2021), neighbor extension-based methods (Pei et al., 2019; Chien et al., 2020) and graph partitioning-based methods (Manaskasemsak et al., 2021; Corizzo & Slenn, 2022; Xu et al., 2021). However, it is a significant challenge to set personalized neighborhood sizes or fraud subgraphs for the different fraudsters. Meanwhile, fraudsters excluded from the neighborhood or fraud subgraph are ignored for aggregation, resulting in valuable information being lost.

Class Imbalance. Another challenge is the class imbalance problem. In general, fraudsters usually make up a fewer proportion than normal users, showing a highly skewed distribution. As shown in Table 2, in the Amazon dataset, 90.5% of the nodes are normal users while only 9.5% are fraudsters and in the Yelpchi dataset, 85.5% of the nodes are normal users while only 14.5% are fraudsters. The class imbalance problem poses a challenge to existing GNN-based methods because the majority of class may dominate the loss function of the GNN, making the trained GNN overfit the majority class (i.e., the normal users) and fail to predict accurately for the minority class (i.e., the fraudsters). Unfortunately, considering that a fraudster may still have the majority of his/her connections to the normal users, the class imbalance exacerbates the graph heterophily with more difficulties for identification. Though some recent works (Liu et al., 2021; Shi et al., 2020; Zhang et al., 2021) have noticed similar challenges, their solutions either use a single-relational balanced sampler or an imbalanced distribution-oriented loss function, both of which fail to consider differences in the prevalence of nodes under different relationships, making them difficult to apply to the problem of fraud detection under multiple relationships.

To address the above challenges, we propose an Imbalanced Graph Structure Learning framework (IGSL) for fraud detection. Specifically, IGSL is composed of four module layers: (1) a multi-relational class-balanced sampler layer. To alleviate the effects of class imbalance, a multi-relational class-balanced nodes sampler is designed with the consideration of nodes' prevalence (i.e., degree) under different relationships and corresponding label class frequency; (2) a graph-independent embedding layer, which is used to roughly pre-process the raw graph to alleviate the graph heterophily by a structure-independent embedding; (3) an iterative graph structure learning layer. We introduce an iteration metric within the graph convolution framework to iteratively construct the global homophilic adjacency matrix, making homophilic nodes connected and heterophilic nodes disconnected from each other; (4) an embedding aggregation layer, which combines the intermediate embeddings to be the final representation of nodes. We summarize the main contributions of our paper as:

- We formulate the graph-based fraud detection problem as an imbalanced heterophily graph node classification task and propose an imbalanced graph structure learning framework to deal with the heterophily and class imbalance problem on graphs.
- An iterative deep graph structure learning method that iteratively constructs the global homophilic adjacency matrix is further developed to make the reconstructed graph structure and node embeddings optimal for fraud detection. A multi-relational class-balanced sampler is proposed to reinforce the learning of the minority class.
- Experiments on two public datasets demonstrate that IGSL outperforms the state-of-the-art baselines.

The rest of our paper is organized as follows. Section 2 reviews relevant studies in the literature. Section 3 details the definition and problem statement. Section 4 describes the model design of our method. Section 5 presents the experimental setup and the corresponding results. Finally, in Sect. 6, we summarize the paper and discuss future work.

2 Related work

Graph-based fraud detection aims at identifying fraudsters from normal users in graph-structured data. Due to the excellent representation capabilities of GNNs, GNNs are widely used in fraud detection. However, classic GNN-based fraud detectors are vulnerable to topology inconsistency and class imbalance problems, and current scholars are dedicated to designing robust GNNs to defend against the problems in graph data. GraphConsis (Liu et al., 2020) is a pioneer work concerning topology inconsistency in fraud detection by implementing dissimilar neighbors filters for nodes based on a pre-defined threshold. CARE-GNN (Dou et al., 2020) adopts a reinforcement learning-based module to extend the neighbor filtering operation to make the thresholds adaptive. PC-GNN (Liu et al., 2021) adopts a label-balanced sampler and neighbors over-sampling/under-sampling strategy to solve the class imbalance problem and topology inconsistency problem. FRAUDRE (Zhang et al., 2021) adopts an imbalance-oriented classification module to solve the class imbalance problem in fraud detection. AO-GNN (Huang et al., 2022) adopts AUC-maximization to resolve the label-imbalance problem for GNNs. Most of the mentioned approaches adopt the simple neighbor filtering-based method or neighbor extension-based method to address the hererophily problem on graphs. Different from them, we formulate the fraud detection problem as an imbalanced graph structure learning task and learn a task-relevant graph structure and node representation on the imbalanced graph to yield optimal results compared to the above methods.

3 Definition and problem statement

In this section, we first give the conception of homophily and heterophily, imbalanced graph, and multi-relational imbalanced graph. Then, we formulate the graph-based fraud detection problem. Furthermore, we summarize important symbols, as shown in Table 1.

3.1 Definition

Definition 1 (Homophily and Heterophily): For a graph, an edge (connection) is homophilic if the two nodes connected by an edge belong to the same class. Otherwise, this edge is heterophilic. A graph consisting of homophilic edges is called a homophilic graph, and a graph consisting of heterophilic edges is called a heterophilic graph. In particular, a fraud graph has both homophilic and heterophilic edges.

Definition 2 (Imbalanced Graph): Given a series of labels $\boldsymbol{Y} = \{y_1, y_2, ..., y_i\}$ in graph $\mathcal{G}$, where i denotes the total number of label classes. We use class imbalance ratio ρ to measure the extent of class imbalance.

$$\rho = \frac{\max_i(|y_i|)}{\min_i(|y_i|)}, \tag{1}$$

where $\max_i(|y_i|)$ and $\min_i(|y_i|)$ return the maximum and minimum class size over all i classes, respectively. If $\rho > 1$, the class with the maximum size is called the majority class,

Table 1 Glossary of notations

Symbol	Definition
$\boldsymbol{\mathcal{G}}; \boldsymbol{\mathcal{V}}; \boldsymbol{\mathcal{E}}; \boldsymbol{\mathcal{A}}; \boldsymbol{\mathcal{X}}$	Graph; Node set; Relation set; Adjacency matrix set; Node attribute vector set
$y_v; \boldsymbol{Y}$	Label for node v; Node label set
$\max_i; \min_i; \rho$	Maximum class size; Minimum class size; Class imbalance ratio
$r; R$	Relation; Total number of relations
$l; L$	GNN layer number; Total number of layers
$\boldsymbol{\mathcal{V}}_{train}; \boldsymbol{\mathcal{V}}_p$	Nodes in the training set; Set of picked nodes
D_v^r	Degree of node v under relation r
$LF(Y(v))$	Frequency of labels for the class $Y(v)$
$\mathcal{E}_r$	Edge under relation r
$\mathcal{A}_r$	Adjacency matrice under relation r
$\boldsymbol{\mathcal{W}}_1$	Learnable weight matrix for MLP
σ	Activation function
$\boldsymbol{f}_i$	Soft assignment matrix
$e_{i,j}^r$	Edge between nodes v_i and v_j under relation r
$\bar{e}_{i,j}^r$	Edge between nodes v_i and v_j under relation r after edge pruning
$\widetilde{e}_{i,j}^r$	Edge between nodes v_i and v_j under relation r after graph-independent embedding
$\tau^-; \tau^+$	Threshold of pruning edges; Threshold of oversampling
$\widetilde{A}_r$	Adjacency matrix under relation r after graph-independent embedding
$A_{r,t}^{(l)}$	Adjacency matrix under relation r at the l-th layer in t-th iteration
$\boldsymbol{H}_{r,t}^{(l)}$	Node embedding under relation r at the l-th layer in t-th iteration
$\Gamma_r^{(l)}$	K-head weighted cosine similarity function
$\boldsymbol{\mathcal{W}}_r^{(l)}$	Learnable weight matrix under relation r in l-th layer
$\boldsymbol{\mathcal{K}}_{r,t}^{(l)}$	Node-Anchor homophilic matrix under relation r at the l-th layer in t-th iteration
δ	Threshold of dynamic stopping

and the class with the minimum size is called the minority class. If ρ is far larger than 1, we call $\boldsymbol{\mathcal{G}}$ an imbalanced graph, otherwise, the graph is balanced.

Definition 3 (Multi-relational Imbalanced Graph): Let $\boldsymbol{\mathcal{G}} = (\boldsymbol{\mathcal{V}}, \boldsymbol{\mathcal{E}}, \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{X}}, \boldsymbol{Y})$ be an undirected multi-relational graph. Specifically, $\boldsymbol{\mathcal{V}} = \{v_1, ..., v_N\}$ denotes a set of nodes; $\boldsymbol{\mathcal{E}} = \{\mathcal{E}_1, \mathcal{E}_2, ..., \mathcal{E}_R\}$ is the edges set of R relationships, where $e_{i,j}^r \in \mathcal{E}_r$ indicates that there is an edge between node i and j under the r-th relationship; $\boldsymbol{\mathcal{A}} = \{A_1, A_2, ..., A_R\}$ denotes the responding adjacency matrix of relationships where $A_{i,j}^r = 1$ if $e_{i,j}^r \in \mathcal{E}_r$; $\boldsymbol{\mathcal{X}} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_N\}$ is the attribute vector of node and $\boldsymbol{x}_i \in \mathbb{R}^d$; $\boldsymbol{Y} = \{y_1, y_2, , ..., y_N\}$ is the label set in which $y_v \in \{0, 1\}$, where 1 represents fraudster and 0 represents normal user, if the class imbalance ratio ρ of $\boldsymbol{\mathcal{G}}$ is far larger than 1, we call $\boldsymbol{\mathcal{G}}$ a multi-relational imbalanced graph.

3.2 Problem statement

Definition 4 (Graph-based Fraud Detection): Given a multi-relational imbalanced graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A}, \mathcal{X}, Y)$ defined in definition 3 where each node has a ground truth label of fraudster or normal user. In this study, the graph-based fraud detection is considered as a transductive semi-supervised binary classification problem on graph $\mathcal{G}$ where both the training and testing samples are accessed during the training phase. The graph-based fraud detector is trained based on the labeled node information and the graph consisting of homophilic edges and heterophilic edges under multiple relationships. The trained model is then used to predict the probability of suspiciousness of unlabeled nodes.

4 Methodology

We first give a brief overview of our approach and then specify the proposed novel method, including a detailed description of each component.

4.1 Overview

In this section, we give details of the proposed framework IGSL. An illustration of the proposed framework is shown in Fig. 2. IGSL is composed of four components: a multi-relational class-balanced sampler module, a graph-independent embedding module, an iterative graph structure learning module, and an aggregation module. Next, we give details of each component.

4.2 Multi-relational class-balanced sampler

We devise a multi-relational class-balanced sampler to pick nodes for training. The key idea lies in incorporating node prevalence in different relationships and class frequency into the

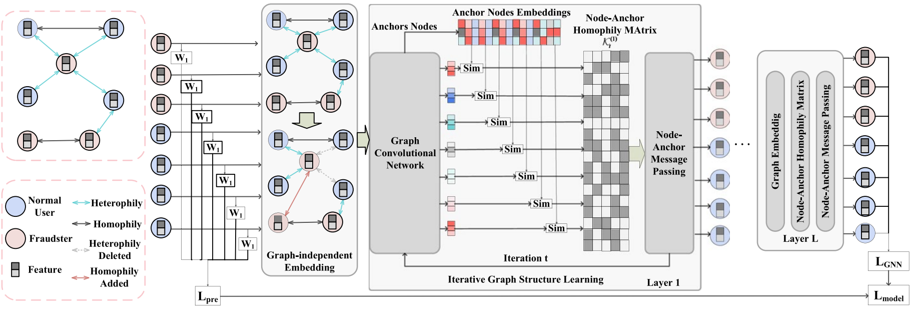

Normal User
Fraudster
Feature
Heterophily
Homophily
Heterophily Deleted
Homophily Added
W_1
Graph-independent Embedding
Anchors Nodes
Anchor Nodes Embeddings
Node-Anchor Homophily MAtrix
Graph Convolutional Network
Sim
Node-Anchor Message Passing
Iteration t
Iterative Graph Structure Learning
Layer 1
Graph Embeddig
Node-Anchor Homophily Matrix
Node-Anchor Message Passing
Layer L
L_{pre}
L_{GNN}
L_{model}

Fig. 2 The figure demonstrates the framework of IGSL. The example graph has seven nodes, the cyan lines indicate heterophilic connections and the dark lines indicate homophilic connections. The nodes in red are fraudsters and blue are normal users. Firstly, a graph-independent embedding module using a multi-layer perceptron to roughly adjust the original structure. Secondly, an iterative graph structure learning module obtains the global homophilic adjacency matrix for fraud detection by performing iterative similarity learning on the node embeddings. Finally, nodes under different relationships are connected to obtain the final representation

sampling process. For node samplers, those of the minority class with more prevalence have a higher sampling probability than the majority class with less prevalence.

Formally, $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A}, \mathcal{X}, Y)$ is a multi-relational imbalanced graph, where $\mathcal{A} = \{A_1, A_2, ..., A_R\}$ denotes adjacency matrix of relationships. For node $v \in \mathcal{V}$, its sampling probability $p_{r,v}$ under relation r is defined as Eq.(2)

$$p_{r,v} \propto \frac{D_{r,v}}{\mathrm{LF}(y_v)}, \tag{2}$$

where $D_{r,v} = \sum_{i \in \mathcal{N}_{r,v}} A_{r,v,i}$ is the degree of node v under relation r which denotes the prevalence of node, $\mathcal{N}_{r,v}$ denotes the neighbors of v under relation r, and $\mathrm{LF}(y_v)$ denotes the frequency of labels for the class y_v. Note that, $D_{r,v}$ means that more "popular" nodes are more likely to selected, and $\mathrm{LF}(y_v)$ means the more "rare" nodes are more likely to be selected. Therefore, the set of picked nodes $\mathcal{V}_p$ is marked as:

$$\mathcal{V}_p = \sum_{v \in \mathcal{V}_{\text{train}}} \sum_{r=1}^{R} \mathrm{Pick}(p_{r,v}), \tag{3}$$

where $\mathcal{V}_{\text{train}}$ denotes the training set, and Pick denotes sampling function.

4.3 Graph-independent embedding

According to Wu et al. (2019), most of the graph heterophily in fraud detection is caused by the extensive illegitimate connections that fraudsters intentionally establish with normal users. A straightforward approach is to remove heterophilic edges and add homophilic edges using a structure-independent approach to the original structure, thus avoiding pitfalls. We apply a graph-agnostic multi-layer perceptron (MLP) to extract class-aware information from the original node attributes:

$$\boldsymbol{h}_v = \sigma(\boldsymbol{x}_v \mathcal{W}_1), \tag{4}$$

where $\boldsymbol{x}_v \in \mathcal{X}$ is the attribute vector of node v, $\mathcal{W}_1 \in \mathbb{R}^{d \times C}$ is the learnable weight matrix for MLP, σ is the activation function, and $\boldsymbol{h}_v$ denotes the output of the final layer of MLP. We can obtain the soft assignment matrix $\boldsymbol{f}_v \in \mathbb{R}^{1 \times C}$ as follows:

$$\boldsymbol{f}_v = soft\max(\boldsymbol{h}_v), \tag{5}$$

where $\boldsymbol{f}_{v,c} \in \boldsymbol{f}_v$ denotes the probability that node v belongs to class C. Let all parameters of the MLP be θ_m, then the optimal θ_m^* is obtained by minimizing the loss of the following predicted labels by the MLP:

$$\theta_m^* = \arg\min_{\theta_m} \mathcal{L}_{\text{pre}} = \arg\min_{\theta_m} - \sum_{v \in \mathcal{V}_p} [y_v \log \boldsymbol{f}_v + (1 - y_v) \log(1 - \boldsymbol{f}_v)]. \tag{6}$$

Since $\boldsymbol{f}_v$ is under the guidance of partially known labels, it can capture class-aware information in attributes. Then, based on the matrix $\boldsymbol{f}_v$, we can calculate the probability that two nodes v_i and v_j belong to the same class:

$$\boldsymbol{S}_{ij} = \boldsymbol{f}_i \cdot \boldsymbol{f}_j^T. \tag{7}$$

Then we prune edges whose scores are below a threshold τ^- under all relationships:

$$\bar{e}^r_{i,j} = \begin{cases} e^r_{i,j} & \boldsymbol{S}_{ij} > \tau^- \\ 0 & \boldsymbol{S}_{ij} < \tau^-, \end{cases} \tag{8}$$

where $e^r_{i,j} \in \mathcal{E}_r$ is an edge between node i and j under the r-th relation. Inspired by PC-GNN(Liu et al., 2021), we pay more attention to the fraud nodes. On the one hand, we remove the heterophilic edges between in the training set, and on the other hand, we oversample the neighbor of fraud nodes. We set a threshold τ^+ to control the oversampling, and fraud nodes with high similarity create links:

$$\widetilde{e}^r_{i,j} = \begin{cases} 0 & y(i) \neq y(j) \\ 1 & y(i) = y(j) = 1 \ and \ \boldsymbol{S}_{ij} > \tau^+ \\ \bar{e}^r_{i,j} & otherwise, \end{cases} \tag{9}$$

where $y(i)$ denotes the label class of node i. It is worth noting that the adjacency matrix $\widetilde{A}_r$ after graph-independent embedding responding to $\widetilde{e}^r_{i,j}$ is estimated based on node features, which are not constrained by the heterophily of networks.

4.4 Iterative graph structure learning

Although the heterophily of the graphs is somewhat alleviated by the graph-independent embedding layer, it is still not possible to eliminate the heterophily as the structural features of the user are ignored. Motivated by IDGL (Chen et al., 2020), we define the problem as an iterative graph structure learning process, learning a global homophilic adjacency matrix that is optimal for fraud detection.

Specifically, we use a GNN to learn t-th iteration node representations $\boldsymbol{H}^{(l)}_{r,t}$ under relation r in t-th iteration by utilizing the message passing scheme to aggregate information from nodes' neighbors:

$$\boldsymbol{H}^{(l)}_{r,t} = \beta \mathrm{Aggr}(\widetilde{A}_{r,ini}, \boldsymbol{H}^{(l)}_{r,t-1}) + (1-\beta)\mathrm{Aggr}(\widetilde{A}^{(l)}_{r,t-1}, \boldsymbol{H}^{(l)}_{r,t-1}), \tag{10}$$

where $\widetilde{A}_{r,ini} = \widetilde{A}_r$ is the adjacency matrix of the initial graph obtained in graph-independent embedding layer, $\widetilde{A}^{(l)}_{r,t-1}$ is the t-1-th homophilic adjacency matrix learned under relation r in layer l, $\boldsymbol{H}^{(l)}_{r,t-1}$ is the t-1-th iteration node embedding under relation r in layer l, $\boldsymbol{H}^{(0)}_{r,0} = \mathcal{X}$ is the raw attribute vector of node, and β is a weight to balance the trade-off of initial graph structure and the homophilic adjacency matrix learned. Specifically, we choose the graph convolutional network (GCN) as the basic GNN. For a relation r, we perform metric learning on the node embedding and obtain homophilic adjacency matrix $\widetilde{A}^{(l)}_{r,t}$ by the learned embedding similarity, where the homophily probability between nodes i and j is obtained by:

$$\widetilde{A}^{(l)}_{r,t}[i,j] = \begin{cases} \Gamma^{(l)}_r(\boldsymbol{h}^{(l)}_{r,t,i}, \boldsymbol{h}^{(l)}_{r,t,j}) & \Gamma^{(l)}_r(\boldsymbol{h}^{(l)}_{r,t,i}, \boldsymbol{h}^{(l)}_{r,t,j}) \geq \varepsilon^{(l)}_r \\ 0 \, otherwise, \end{cases} \tag{11}$$

where $\varepsilon^{(l)}_r \in [-1, 1]$ is the threshold that controls the sparsity, $\boldsymbol{h}_{r,t,i} \in \boldsymbol{H}_{r,t}$ is the embedding of node i under relation r in t-th interation, and $\Gamma^{(l)}_r$ is a K-head weighted cosine similarity function defined as:

$$\Gamma^{(l)}_{r,ij} = \frac{1}{K}\sum_{p=1}^{K} \cos(\mathcal{W}^{(l)}_{r,p} \odot \boldsymbol{h}_{r,t,i}, \mathcal{W}^{(l)}_{r,p} \odot \boldsymbol{h}_{r,t,j}), \tag{12}$$

where $\odot$ denotes the Hadamard product, $\mathcal{W}^{(l)}_{r}$ is the learnable weight matrix that weights the importance of different dimensions of the feature vectors under relation r in l-th layer.

*Node-anchor homophilic matrix*However, Eq. (12) calculates the similarity score for all pairs of graph nodes under all relationships in all iterations with a computational complexity of $O(N^2 * r * t * K)$, which is difficult to apply to fraud detection in large graphs. Inspired by anchor-based methods, we design an anchor-based similarity learning which learns a node-anchor matrix $\mathcal{K} \in \mathbb{R}^{(N\times s)}$ with a computational complexity of O(ns) where s is the number of anchors. Specifically, we randomly select a set of $s \in \mathcal{V}$ anchors nodes, where the number of s is usually much smaller than N. Thus, Eq. (11) can be rewritten as follows:

$$\mathcal{K}^{(l)}_{r,t}[i,k] = \begin{cases} \Gamma^{(l)}_{r}(\boldsymbol{h}^{(l)}_{r,t,i}, \boldsymbol{h}^{(l)}_{r,t,k}) & \Gamma^{(l)}_{r}(\boldsymbol{h}^{(l)}_{r,t,i}, \boldsymbol{h}^{(l)}_{r,t,k}) \geq \varepsilon^{(l)}_{r} \\ 0\, otherwise, \end{cases} \tag{13}$$

where k is the anchor node. Similarly, we adopt a threshold $\varepsilon^{(l)}_{r} \in [-1, 1]$ to control the sparsity of the node-anchor graph $\mathcal{K}^{(l)}_{r,t}$.

Node-anchor message passing According to the stationary Markov random walk theory, the homophilic adjacency matrix $A^{(l)}_{r,t-1}$can be fully recovered by the node-anchor homophilic matrix $\mathcal{K}^{(l)}_{r,t-1}$.

Further, we decompose graph embedding based on the learned adjacency matrix into two steps to reduce the computational complexity. 1): compute the message propagation using the anchor-node matrix $\mathcal{K}^{(l)T}_{t-1}$ and 2): compute the message propagation using the node-to-anchor matrix $\mathcal{K}^{(l)}_{t-1}$. Thus, the GCN($A^{(l)}_{r,t-1}, \boldsymbol{H}^{(l)}_{r,t-1}$) could be explained as follows,

$$\begin{aligned} \boldsymbol{H}^{(l)}_{r,t-1} &= \Lambda^{-1}\mathcal{K}^{(l)T}_{r,t-1}\boldsymbol{H}^{(l)}_{r,t-1} \\ \mathrm{GCN}(A^{(l)}_{t-1}, \boldsymbol{H}^{(l)}_{r,t-1}) &= \Delta^{-1}\mathcal{K}^{(l)}_{r,t-1}\boldsymbol{H}^{(l)}_{r,t-1}, \end{aligned} \tag{14}$$

where $\Delta_{kk} = \sum_{i=1}^{n} \mathcal{K}^{(l)}_{ik}$ and $\Lambda_{ii} = \sum_{k=1}^{s} \mathcal{K}^{(l)}_{ik}$.

Dynamic stopping strategy we define the dynamic stopping strategy for graph structure learning as:

$$\frac{|\mathcal{K}^{(l)}_{t} - \mathcal{K}^{(l)}_{(t-1)}|^2_F}{|\mathcal{K}^{(l)}_{t}|^2_F} > \delta, \tag{15}$$

where $|\cdot|_F$ denotes the Frobenius norm of a matrix, δ is the threshold of dynamic stopping. This means that graph structure learning converges in the two most recent iterations and indicates that an optimal homophilic adjacency matrix for fraud detection is found. After dynamic stopping, the current node embeddings $\boldsymbol{H}^{(l)}_{r,t}$ are used for the fraud detection.

4.5 Embedding aggregation

After the dynamic stopping, we have obtained the optimal node embedding $\boldsymbol{H}^{(l)}_{r} \in \mathbb{R}^{(N\times d_l)}$ at layer l under relation r, where $r = 1, ...R$, $l = 1, ...L$ and L is the number of layers. We adopt concatenation to aggregate the embeddings of nodes under different layers and different relationships, which is illustrated as Eq. (16) and $\boldsymbol{W}^{l} \in \mathbb{R}^{(d_l \times (d_{l-1}+R \cdot d_l))}$ is the learnable weight matrix.

$$\boldsymbol{H}^{(l)} = \mathrm{ReLU}(\boldsymbol{W}^{(l)}(\boldsymbol{H}^{(l-1)} \oplus \boldsymbol{H}^{(l)} \oplus \boldsymbol{H}_2^{(l)} \cdots \boldsymbol{H}_R^{(l)})). \tag{16}$$

The loss function of fraud detection is defined as follows:

$$\begin{aligned} \mathcal{L}_{\text{GNN}} &= -\sum_{v \in \boldsymbol{\mathcal{V}}_p} [y_v \log z_v + (1-y_v)\log(1-z_v)] \\ z_v &= \mathrm{MLP}(\boldsymbol{H}_v^{(l)}), \end{aligned} \tag{17}$$

```
Algorithm 1 Learning Strategy of the IGSL
Input: $\boldsymbol{\mathcal{G}}$: A multi-relational imbalanced graph; $A_r$: Raw adjacency matrix
    under relation $r$; $\boldsymbol{\mathcal{V}}_{\text{train}}$ : Set of training nodes; $\boldsymbol{\mathcal{V}}_p$ : Set of picked nodes
    in each epoch; $N_{\text{epoch}}$: Number of training epochs; $L$: Number of layers;
    $N_{\text{batch}}$: Number of training batch size and $R$: Number of relations.
Output: Vector representations of nodes in $\boldsymbol{\mathcal{V}}_{train}$.
Initialization $p_{r,v} \propto \frac{D_{r,v}}{\mathrm{LF}(y_v)}$, $v \in \boldsymbol{\mathcal{V}}_{\text{train}}$
for $e \leftarrow 1, ..., N_{\text{epoch}}$ do
    Pick $\boldsymbol{\mathcal{V}}_p$ nodes w.r.t. Eq.(3);
    Decide the number of training batches $B = \lceil \frac{\boldsymbol{\mathcal{V}}_p}{N_{\text{batch}}} \rceil$;
    for $b \leftarrow 1, ..., B$ do
        Roughly pre-process $A_r$ to obtain $\widetilde{A}_r$ w.r.t. Eq.(8) and Eq.(9);
        Obtain $\mathcal{L}_{\text{pre}}$ w.r.t. Eq.(6);
        for $l \leftarrow 1, ..., L$ do
            for $r \leftarrow 1, ..., R$ do
                Update $StopCond$ w.r.t. Eq.(15);
                while $t \leqslant T$ or $StopCond$ do
                    Update $\boldsymbol{H}_{r,t}^{(l)}$ w.r.t. Eq.(10);
                    Update $\boldsymbol{\mathcal{K}}_{r,t}^{(l)}$ w.r.t. Eq.(13);
                    Update $t = t + 1$;
                end while
            end for
            Obtain $\boldsymbol{H}^{(l)}$ w.r.t. Eq.(16);
        end for
        Obtain $\mathcal{L}_{\text{GNN}}$ w.r.t. Eq.(17);
        Obtain $\mathcal{L}_{\text{Model}}$ w.r.t. Eq.(18);
    end for
end for
return $\boldsymbol{H}_v^{(L)}$, $v \in \boldsymbol{\mathcal{V}}_{\text{train}}$.
```

Algorithm 1 Learning strategy of the IGSL

where $\mathcal{L}_{\mathrm{GNN}}$ refers to the cross-entropy loss between the prediction result and the ground truth, MLP refers to the multi-layer perceptron, y_v is either equal to 1 or 0 for the ground truth, and z_v refers to the identification probability for the $\boldsymbol{H}_v^{(l)}$ outputted by MLP.

The overall loss function of model is formulated as Eq. (18), where α is weights to balance the importance of different losses, $||\theta||^2$ is the regularization term to avoid over-fitting, and λ is the regularization coefficient.

$$\mathcal{L}_{\mathrm{Model}}=\mathcal{L}_{\mathrm{GNN}}+\alpha\mathcal{L}_{\mathrm{pre}}+\frac{\lambda}{2}||\theta||^2. \tag{18}$$

The overall training algorithm is summarized in Algorithm 1.

5 Experimental evaluation

In this section, we investigate the effectiveness and robustness of the proposed IGSL model for graph-based opinion fraud detection tasks. Our goal is to answer the following questions.

- **RQ1**: Does IGSL outperform the state-of-the-art methods for graph-based fraud detection?
- **RQ2**: Does IGSL benefit from all four modules?
- **RQ3**: How robust is IGSL on different heterophily and class-imbalance rates?
- **RQ4**: How efficient is IGSL to run?
- **RQ5**: How does the hyperparameters affect the performance of IGSL ?

5.1 Experimental setup

Dataset. We adopt two multi-relational opinion fraud detection datasets **Amazon** (McAuley & Leskovec, 2013) and **YelpChi** (Rayana & Akoglu, 2015) to validate the performance of IGSL. The statistical information of these two datasets is shown in Table 2. The Amazon dataset includes both legitimate and fraud reviews under the musical instrument category. In the Amazon dataset, nodes are users with 25-dimensional features and three manually defined relational graphs (i.e., meta-paths) are contained: 1) Relation U-P-U connects users who have reviewed more than one same product; 2) Relation U-S-U connects users who have more

Table 2 Statistics of multi-relational opinion fraud detection datasets

Dataset	Node (Fraudster%)	ρ	Relations	Avg.Label Similarity	Avg.feature Similarity
Amazon		9.9	U-P-U	0.19	0.61
	11,944		U-S-U	0.04	0.64
	(9.5%)		U-V-U	0.03	0.71
			ALL	0.05	0.65
YelpChi		5.8	R-U-R	0.90	0.83
	45,954		R-S-R	0.05	0.77
	(14.5%)		R-T-R	0.05	0.79
			ALL	0.07	0.77

than one same star rating in a week; 3) Relation U-V-U connects users with top 5% of mutual review TF-IDF similarities among all users. Similarly, the YelpChi dataset includes both legitimate and fraudulent reviews of restaurants and hotels. In the YelpChi dataset, nodes are reviews with 32-dimensional features. Analogous to the Amazon dataset, this dataset includes three manually defined relational graphs: 1) Relation R-U-R connects reviews posted by the same user; 2) Relation R-S-R connects reviews with the same star rating (1–5 stars) under the same product; 3) Relation R-T-R connects reviews posted under the same product in the same month. All graphs under different relationships are merged together to form the single-relation ALL.

5.2 Baselines

We compare several GNN-based fraud detection algorithms to demonstrate the effectiveness and superiority of our method in identifying fraudsters.

Classical GNN-based model: GCN (Kipf & Welling, 2017), GAT (Veličković et al., 2018), DR-GCN (Shi et al., 2020), GraphSAGE (Hamilton et al., 2017), GraphSAINT (Zeng et al., 2019).

Classical graph structure learning model: IDGL (Chen et al., 2020) and Pro-GNN (Jin et al., 2020).

State-of-the-art GNN-based fraud method: GraphConsis (Liu et al., 2020), CARE-GNN (Dou et al., 2020), PC-GNN (Liu et al., 2021), FRAUDRE (Zhang et al., 2021), AO-GNN (Huang et al., 2022).

Among those baselines, GCN, GAT, DR-GCN, GraphSAGE, GraphSAINT, IDGL, and Pro-GNN are run on the single-relational graph (i.e., relation ALL in Table 2, where all relationships are merged while GraphConsis, CARE-GNN, PC-GNN, FRAUDRE, AO-GNN, and IGSL are run on the multi-relational graph.

5.3 Experimental setting and implementation

We use the Adam optimizer to optimize the parameters of the IGSL with the learning rate set to 0.01, and the weight decay is 0.001. In the IGSL, $N_{\text{epoch}} = 150$, $L = 1$, batch size N_{batch} set to 256 (Amazon) and 1024 (YelpChi), $\beta = 0.5$, $\epsilon_r^{(l)}$ (ϵ for all relations and layers) set to 0.5 (Amazon) and 0 (YelpChi), $\tau^- = 0.2$, and τ^+ is decided by the top-k distance of the minority class, where k equals half the average neighbor size of the minority class. α and λ in the overall loss function are set to 2 and 0.002, respectively. For the training set and test set split, we follow FRAUDRE (Zhang et al., 2021) with the same random seed. For baselines of state-of-the-art GNN fraud method, we use the default hyperparameter settings according to published papers. For the remaining baselines (i.e., classical GNN models), we refer to the hyperparameters provided by previous research works, such as PC-GNN. Additionally, IGSL is implemented in torch 1.10.1 with Python 3.8 and all experiments are run on Ubuntu 20.04.1 LTS server with Cuda 11.4. Our source code is available at https://github.com/Ling-Fei-Ren/IGSL.git

For class imbalance classification, the evaluation metrics should be unbiased for any class (Luque et al., 2019). Like previous work (Liu et al., 2021), we utilize ROC-AUC (AUC), F1-macro, and GMean to evaluate the overall performance of all models.

Table 3 Performance (%) under different percentages of training data. F1 (abbr. F1-macro)

	Method	40%			30%			20%			10%		
		AUC	F1	GMean	AUC	F1	GMean	AUC	F1	GMean	AUC	F1	GMean
Amazon	GCN	83.67	64.10	57.22	82.34	63.23	57.10	81.49	62.78	56.38	79.20	60.23	55.49
	GAT	81.02	64.64	66.75	82.23	63.78	57.01	80.89	63.23	55.87	80.12	61.33	55.89
	DR-GCN	82.95	64.88	79.63	81.89	63.21	79.23	80.22	63.01	77.98	79.87	62.33	76.65
	GraphSAGE	75.89	64.16	59.49	75.12	65.56	59.12	73.23	64.38	58.34	71.21	60.23	54.78
	GraphSAINT	87.00	74.23	74.78	86.34	73.56	72.87	84.56	71.90	70.00	82.45	71.33	69.78
	IDGL	19.14	47.50	0	23.87	47.50	0	17.61	47.50	0	22.48	47.50	0
	Pro-GNN	73.23	47.63	0	73.61	47.06	0	73.07	47.50	0	72.34	46.88	0
	GraphConsis	87.41	75.12	76.77	87.65	75.76	75.32	86.88	74.21	74.32	85.34	71.28	72.98
	CARE-GNN	94.97	90.20	90.05	91.89	87.78	87.35	93.71	88.44	89.09	94.05	88.92	87.38
	PC-GNN	95.86	89.56	90.30	95.50	88.04	86.00	95.20	86.39	87.71	94.61	87.61	88.21
	FRAUDRE	93.36	90.45	89.51	94.52	88.27	87.94	94.45	88.95	89.49	92.87	83.19	87.98
	AO-GNN	96.39	89.21	90.96	95.99	88.98	**90.99**	95.15	87.58	**90.65**	94.82	82.36	87.67
	IGSL (ours)	**97.76**	**91.45**	**91.63**	**96.84**	**90.10**	89.11	**96.14**	**90.58**	89.05	**95.40**	**89.86**	**88.54**

Table 3 (continued)

	Method	40%			30%			20%			10%		
		AUC	F1	GMean	AUC	F1	GMean	AUC	F1	GMean	AUC	F1	GMean
Yelpchi	GCN	59.83	56.20	43.66	58.99	55.76	44.67	57.10	54.23	42.67	56.48	52.10	40.22
	GAT	57.15	48.79	16.59	57.01	49.01	16.88	56.88	47.89	15.49	55.23	45.34	13.21
	DR-GCN	59.21	55.23	40.38	58.11	56.21	41.23	57.01	54.92	39.89	56.21	54.21	38.32
	GraphSAGE	54.39	44.05	25.89	55.67	45.23	26.10	53.88	42.88	24.90	51.39	40.10	22.00
	GraphSAINT	68.99	58.22	58.29	66.66	56.20	57.89	66.09	56.10	55.91	64.33	53.00	54.00
	IDGL	50.92	46.08	0	50.03	45.08	0	50.99	46.08	0	51.54	46.08	0
	Pro-GNN	59.88	48.02	0	58.10	45.00	0	59.11	47.00	0	59.88	48.02	0
	GraphConsis	69.83	58.70	58.57	67.12	57.98	58.21	66.43	56.76	57.32	64.32	55.43	54.02
	CARE-GNN	77.71	62.62	71.19	76.97	65.07	64.81	76.07	59.61	69.84	75.28	61.12	67.77
	PC-GNN	81.78	64.00	73.95	81.21	64.07	73.46	78.48	65.53	71.02	77.14	66.71	66.48
	FRAUDRE	76.99	64.21	67.28	76.30	53.41	69.45	75.11	59.88	67.90	73.33	58.15	64.87
	AO-GNN	88.05	70.42	81.34	85.25	**72.39**	75.77	83.03	42.68	34.47	81.16	63.15	66.81
	IGSL(ours)	**90.57**	**72.28**	**82.60**	**86.24**	64.25	**77.31**	**84.56**	**68.79**	**76.71**	**82.13**	**68.32**	**74.73**

5.4 Performance comparison (RQ1)

To answer RQ1, we compare the proposed framework IGSL with the baselines in the opinion fraud detection task. The corresponding AUC, F1-macro, and GMean on Amazon and YelpChi are reported in Table 3. We can make following observations:

Firstly, IGSL outperforms the baselines in terms of AUC, F1-macro, and GMean metrics at different training percentages. For example, when the training percentage is set to 40%, IGSL achieves performance improvements of 1.42%, 2.51%, and 0.73% on the Amazon dataset in terms of AUC, F1-macro, and GMean compared with the state-of-the-art baseline model. On the Yelpchi dataset, the performance gap is 2.86%, 2.64%, and 1.54%, respectively. This is because IGSL employs an iterative graph structure learning process to learn a global homophilic graph that is optimal for downstream fraud detection tasks, which could be a better graph input for GNNs to learn better node embeddings. In contrast, neighbor filtering-based methods in state-of-the-art baselines tend to fall into local optima. In addition, a multi-relational balanced sampler can further alleviate the problem of graph heterophily increasing due to class imbalance by considering the prevalence of nodes under different relationships. Therefore, the proposed model exhibits superior performance compared to all baselines. However, it is worth noting that in some cases the IGSL model does not achieve the best performance. For example, when the training ratio is set to 30%, the AO-GNN model achieved a higher GMean score on the Amazon dataset compared to IGSL. This is due to the fact that AO-GNN employs both classifier parameter search and edge pruning policy search methods, respectively. The former mitigates the problem of class imbalance, and the latter policy searching is designed for graph noise removal.

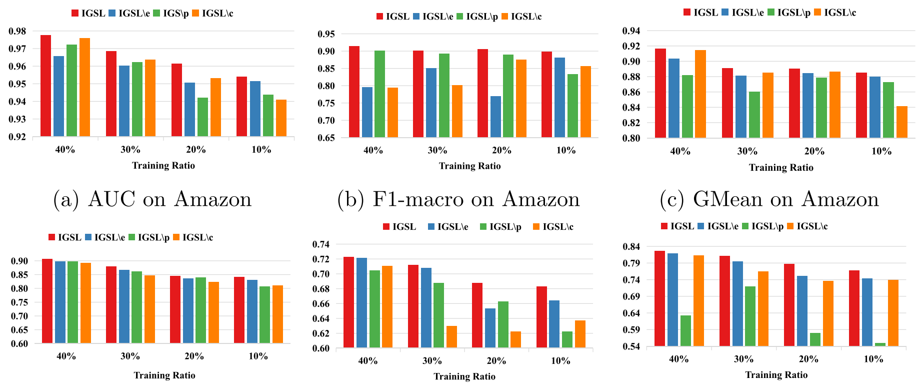
Secondly, GraphConsis, CARE-GNN, PC-GNN, FRAUDRE, and AO-GNN are five state-of-the-art multi-relational graph-based fraud detection methods, in which GraphConsis and CARE-GNN only focus on the heterophily problems while PC-GNN and FRAUDRE focus on both graph heterophily and the class imbalance at the same time. PC-GNNN, FRAUDRE and AO-GNN perform better than GraphConsis and CARE-GNN in all these metrics, which indicates that solving the class imbalance problem is helpful for fraud detection. Meanwhile, PC-GNN and AO-GNN performs better than FRAUDRE, indicating the contribution of modifying the local structure to fraud detection.

Thirdly, IDGL and Pro-GNN are two representative graph structure learning on the balanced graph. However, they perform worse than IGSL, which indicates that classical graph structure learning models are not suitable for fraud detection where classes are imbalanced.

Fourthly, GCN, GAT, DR-GNN, GraphSAGE, and GraphSAINT are five classical GNN-based methods implemented in the single-relational graph ALL. The performance of these methods is generally worse than that of GraphConsis, CARE-GNN, PC-GNN, and FRAUDRE which run in a multi-relational graph. The reason for this is, on the one hand, none of these methods take any measures to alleviate the class imbalance. On the other hand, a single-relational graph loses multi-relational profiles, leading to the aggravation of graph heterophily.

5.5 Ablation study (RQ2)

To answer RQ2, we conduct ablation experiments to verify the contribution of each module. The three variants are (1) *IGSL* $\backslash p$: which does not include the multi-relational class-balanced sampler module, so the nodes are sampled with the same probability. (2) *IGSL*



(a) AUC on Amazon (b) F1-macro on Amazon (c) GMean on Amazon

(b) AUC on Yelpchi (b) F1-macro on Yelpchi (c) GMean on Yelpchi

Fig. 3 Ablation study

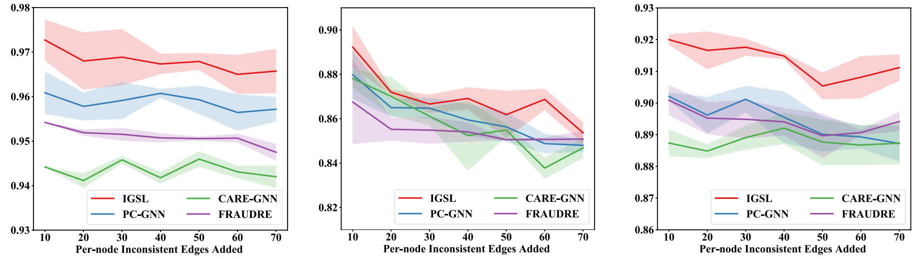
$\backslash e$: which does not include the graph-independent embedding module, so the iterative graph structure learning module is performed on the raw graph directly. (3) $IGSL\ \backslash \mathbf{c}$: which switches the graph structure learning to GCN so the neighbor aggregated as the raw graph. As shown in Fig. 3, IGSL is superior to the above three variants w.r.t. all evaluation metrics as the percentage of training samples varies from 40% to 10%. Specially, we can observe:

The performance of $IGSL\ \backslash p$ is comparably worse than IGSL, which shows the necessity of a multi-relational class-balanced sampler, which is better at finely modeling the importance of different classes under different relationships than a simple single-relational imbalanced sampler.

The performance of $IGSL\ \backslash c$ is comparably worse than IGSL, which demonstrates that graph structure learning has a contribution to solving the graph heterophily problem.

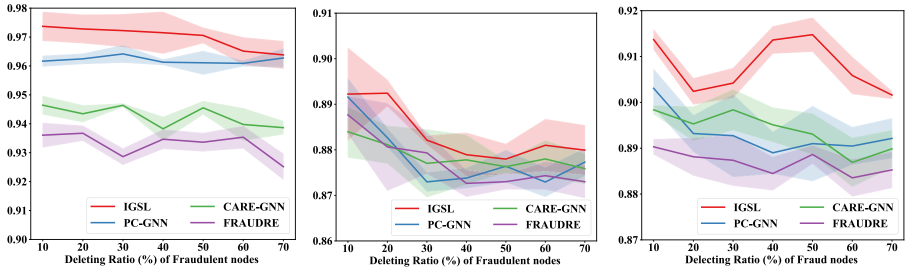
5.6 Robustness comparison (RQ3)

To answer RQ3, we evaluate the robustness of IGSL in defending against graph heterophily and class imbalance. We only report results on Amazon, since similar patterns are observed in YelpChi.



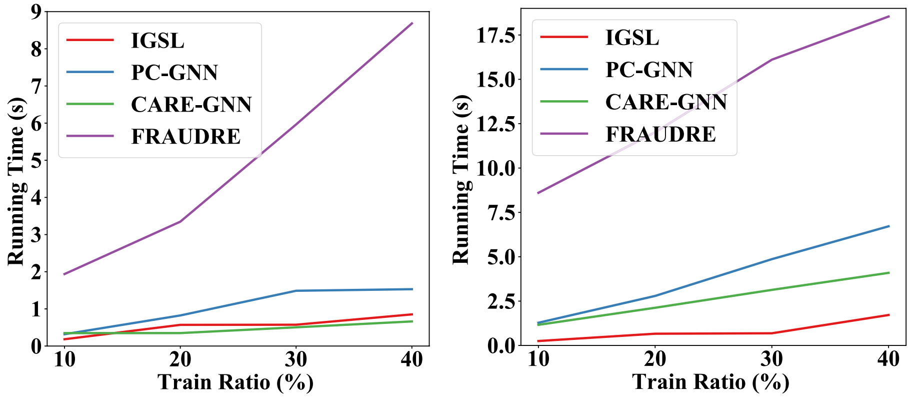
(a) AUC (b) F1-macro (c) GMean

Fig. 4 Robustness for graph heterophily



(a) AUC (b) F1-macro (c) GMean

Fig. 5 Robustness for class imbalance



(a) Amazon (b) YelpChi

Fig. 6 Time efficiency analysis on Amazon and YelpChi

Robustness for graph heterophily. To evaluate the robustness of IGSL to graph heterophily, we vary the heterophilic edges proportion in the multi-relational graph. We randomly select part of fraud nodes (100 nodes for example) and add heterophilic edges with 10, 20, 30, 40, 50, 60, and 70 normal users with highly similar features (feature similarity > 0.8) to each fraud node in the raw graphs. Every experiment is conducted 10 times and the average results are presented in Fig. 4, IGSL achieves better performance compared to PC-GNN, CARE-GNN, and FRAUDRE. We speculate that the local neighbors filtering strategy cannot completely filter the heterophilic edges, leading to sub-optimal results, while the graph structure learning method obtains the global homophily adjacency matrix for fraud detection in an iterative manner, which is robust to heterophily.

Robustness for class imbalance. To evaluate the robustness of IGSL to class imbalance, we vary the class imbalance rate by randomly deleting 10%, 20%, 30%, 40%, 50%, 60%, and 70% of the fraud nodes from the training set. Every experiment is conducted 10 times and the average results are presented. As shown in Fig. 5, IGSL achieves better results for different class imbalance ratios compared to PC-GNN, CARE-GNN, and FRAUDRE.

5.7 Time efficiency (RQ4)

To answer RQ4, we evaluate the run-time performance of IGSL and the baselines and record the average training time per epoch, with the proportion of training data varying from 10% to 40%. As shown in Fig. 6, IGSL has obvious advantages in running speed compared to the multi-relational graph-based fraud detection algorithms (e.g., CARE-GNN, PC-GNN, and FRAUDRE).

5.8 Hyper-parameter sensitivity (RQ5)

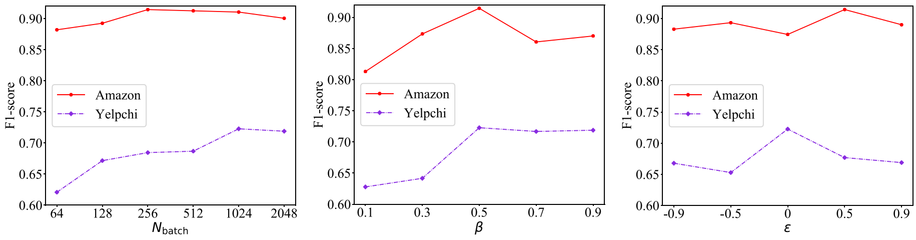
IGSL involves several hyperparameters. In this section, we study the hyperparameters that we consider critical for IGSL (i.e., batch size N_{batch}, balance weight β, and sparsity threshold ε) and investigate the robustness of IGSL to various parameter settings. For a fair comparison, we only vary the value of the investigated parameter with all other parameters fixed.

5.8.1 Effect of batch size N_{batch}

To investigate the effect of batch size N_{batch} on the final detection, we varied the value of batch size N_{batch} in the range of [64, 2048]. The results are shown in Fig. 7a, and it can be observed that the performance improves as the batch size increases. In fact, too small a batch size leads to instability in network convergence. Too large a batch size means less training time but requires large running memory requirements and can easily fall into local optimum. To make a tradeoff between performance and training time, we finally set the batch sizes N_{batch} to 512 and 1024 for Amazon and Yelpchi, respectively.

5.8.2 Effect of balance weight β

To investigate the effect of the balance weight β on the final detection, we varied the value of the balance weight β in the range of [0.1, 0.9]. Specifically, β can influence the importance of the initialized adjacency matrix in iterative graph structure learning. A smaller β indicates a smaller weight of the initialized adjacency matrix and vice versa. The results are shown in Fig. 7b, and it can be observed that the detection performance increases with β increasing, which means that the increased β can capture the useful information in the original graph structure. When $\beta > 0.5$, the detection performance fluctuates, which means



(a) F1-score with varying N_{batch}

(b) F1-score with varying β

(c) F1-score with varying ϵ

Fig. 7 Hyper-parameter sensitivity

a too-large β further captures the noise in the original graph structure. Generally, the performance F1-score is best when the balance weight β is 0.5 for both datasets.

5.8.3 Effect of sparsity threshold ϵ

To investigate the effect of the sparse threshold $\epsilon_r^{(l)}$ (ϵ for all relationships and layers in the experiment) on the final detection, we vary the value of the sparse threshold ϵ in the range of [− 1, 1]. Specifically, ϵ can affect the sparsity of the learned adjacency matrix. A smaller ϵ indicates a smaller sparsity of the learned adjacency matrix and more computational resources required, and vice versa. The results are shown in Fig. 7c, where the performance of Amazon is best when $\epsilon = 0.5$ and the performance of Yelpchi is best when $\epsilon = 0$.

6 Conclusion and future work

In this paper, we propose an imbalanced graph structure learning framework called IGSL to solve the graph heterophily and class imbalance problems in fraud detection. To solve the graph heterophily problem, we further develop an iterative graph structure learning module to iteratively aggregate global homophilic neighbors to learn a homophilic graph structure, which helps to make nodes with the same class have similar embedding representations while nodes with different classes have different embedding representations. For the class imbalance problem, a multi-relational class-balanced sampler is designed that considers the nodes' prevalence under different relationships and corresponding class frequencies, which helps the model to eliminate the bias towards the major class (i.e., normal users) and focus more on the valuable minority class (i.e., fraudsters). Extensive experiments on two public fraud datasets demonstrate the effectiveness of our approach.

For future work, there are some research directions worth studying: (1) Fraud detection on dynamic graphs. In fact, user behaviors and social structures keep changing over time, and some studies (Rao et al., 2022; Jiang et al., 2022) have demonstrated that the dynamic evolution patterns of fraudsters differ significantly from those of normal users, and that fraud can be better modeled using dynamic graph-based methods. (2) Fraud detection based on non-manually defined meta-paths. Manually defined meta-paths require same priori knowledge from domain experts, which makes wide application limited. However, some current works (Hussein et al., 2018; Petković et al., 2022) have made it possible to extract meta-paths automatically, or without the need to define them in advance. They inspire us to study fraud detection based on non-manually defined meta-paths in future work.

Author contributions Authors' contributions follow the authors' order convention.

Funding The work is supported by the National Nature Science Foundation of China (No.U22A2035, No.U1803262), National Social Science Foundation of China (No.19ZDA113), Application Foundation Frontier Project of Wuhan Science and Technology Bureau (No.2020010601012288) and National Nature Science Foundation of China (No.U1736206).

Data availibility statement The datasets are available from the corresponding author on reasonable request.

Declarations

Conflicts of interest The authors of this work declare no confict of interest.

Ethics approval Not applicable.

Consent to participate Yes.

Consent for publication Yes.

Code availability The source code of the current work is available from the corresponding author on reasonable request.

References

Abu-El-Haija, S., Perozzi, B., Kapoor, A., Alipourfard, N., Lerman, K., Harutyunyan, H., Ver Steeg, G., & Galstyan, A. (2019). Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *International conference on machine learning* (pp. 21–29). PMLR.

Chen, Y., Wu, L., & Zaki, M. (2020). Iterative deep graph learning for graph neural networks: Better and robust node embeddings. *Advances in Neural Information Processing Systems, 33*, 19314–19326.

Chien, E., Peng, J., Li, P., & Milenkovic, O. (2020). Adaptive universal generalized pagerank graph neural network. In *International conference on learning representations*.

Corizzo, R., & Slenn, T. (2022). Distributed node classification with graph attention networks. In *2022 IEEE international conference on big data (big data)* (pp. 3720–3725). IEEE.

Dou, Y., Liu, Z., Sun, L., Deng, Y., Peng, H., & Yu, P. S. (2020). Enhancing graph neural network-based fraud detectors against camouflaged fraudsters. In *Proceedings of the 29th ACM international conference on information & knowledge management* (pp. 315–324).

Ge, S., Ma, G., Xie, S., & Philip, S. Y. (2018). Securing behavior-based opinion spam detection. In *2018 IEEE international conference on big data (big data)* (pp. 112–117). IEEE

Hamilton, W., Ying, Z., & Leskovec, J. (2017). Inductive representation learning on large graphs. *Advances in Neural Information Processing Systems 30*.

Huang, M., Liu, Y., Ao, X., Li, K., Chi, J., Feng, J., Yang, H., & He, Q. (2022). Auc-oriented graph neural network for fraud detection. In *Proceedings of the ACM web conference 2022* (pp. 1311–1321).

Hussein, R., Yang, D., & Cudré-Mauroux, P. (2018). Are meta-paths necessary? Revisiting heterogeneous graph embeddings. In *Proceedings of the 27th ACM international conference on information and knowledge management* (pp. 437–446).

Jiang, Y., Liu, G., Wu, J., & Lin, H. (2022). Telecom fraud detection via Hawkes-enhanced sequence model. *IEEE Transactions on Knowledge and Data Engineering*.

Jin, W., Ma, Y., Liu, X., Tang, X., Wang, S., & Tang, J. (2020). Graph structure learning for robust graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 66–74).

Kaghazgaran, P., Alfifi, M., & Caverlee, J. (2019). Wide-ranging review manipulation attacks: Model, empirical study, and countermeasures. In *Proceedings of the 28th ACM international conference on information and knowledge management* (pp. 981–990).

Kipf, T.N., & Welling, M. (2017) Semi-supervised classification with graph convolutional networks. In *International conference on learning representations*.

Liu, Y., Ao, X., Qin, Z., Chi, J., Feng, J., Yang, H., & He, Q. (2021). Pick and choose: a GNN-based imbalanced learning approach for fraud detection. In *Proceedings of the Web Conference 2021* (pp. 3168–3177).

Liu, Z., Dou, Y., Yu, P.S., Deng, Y., & Peng, H. (2020). 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* (pp. 1569–1572).

Liu, C., Sun, L., Ao, X., Feng, J., He, Q., & Yang, H. (2021) Intention-aware heterogeneous graph attention networks for fraud transactions detection. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining* (pp. 3280–3288).

Luque, A., Carrasco, A., Martín, A., & de Las Heras, A. (2019). The impact of class imbalance in classification performance metrics based on the binary confusion matrix. *Pattern Recognition, 91*, 216–231.

Manaskasemsak, B., Tantisuwankul, J., & Rungsawang, A. (2021). Fake review and reviewer detection through behavioral graph partitioning integrating deep neural network. *Neural Computing and Applications* (pp. 1–14).

McAuley, J.J., & Leskovec, J. (2013) From amateurs to connoisseurs: Modeling the evolution of user expertise through online reviews. In *Proceedings of the 22nd international conference on World Wide Web* (pp. 897–908).

Pei, H., Wei, B., Chang, K.C.-C., Lei, Y., & Yang, B. (2019). Geom-GCN: geometric graph convolutional networks. In *International conference on learning representations*.

Petković, M., Ceci, M., Pio, G., Škrlj, B., Kersting, K., & Džeroski, S. (2022). Relational tree ensembles and feature rankings. *Knowledge-Based Systems, 251*, 109254.

Rao, S.X., Lanfranchi, C., Zhang, S., Han, Z., Zhang, Z., Min, W., Cheng, M., Shan, Y., Zhao, Y., & Zhang, C. (2022). Modelling graph dynamics in fraud detection with" attention". *International conference on learning representations*.

Rayana, S., & Akoglu, L. (2015) Collective opinion spam detection: Bridging review networks and metadata. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 985–994).

Shi, M., Tang, Y., Zhu, X., Wilson, D., & Liu, J. (2020). Multi-class imbalanced graph convolutional network learning. In *Proceedings of the twenty-ninth international joint conference on artificial intelligence (IJCAI-20)*.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2018). Graph attention networks. In *International conference on learning representations*.

Van Vlasselaer, V., Eliassi-Rad, T., Akoglu, L., Snoeck, M., & Baesens, B. (2017). Gotcha! network-based fraud detection for social security fraud. *Management Science, 63*(9), 3090–3110.

Wang, D., Lin, J., Cui, P., Jia, Q., Wang, Z., Fang, Y., Yu, Q., Zhou, J., Yang, S., & Qi, Y. (2019). A semi-supervised graph attentive network for financial fraud detection. In *2019 IEEE international conference on data mining (ICDM)* (pp. 598–607). IEEE.

Wu, H., Wang, C., Tyshetskiy, Y., Docherty, A., Lu, K., & Zhu, L. (2019). Adversarial examples for graph data: Deep insights into attack and defense. In *Proceedings of the 28th international joint conference on artificial intelligence* (pp. 4816–4823)

Xu, H., Duan, Z., Wang, Y., Feng, J., Chen, R., Zhang, Q., & Xu, Z. (2021). Graph partitioning and graph neural network based hierarchical graph matching for graph similarity computation. *Neurocomputing, 439*, 348–362.

Zeng, H., Zhou, H., Srivastava, A., Kannan, R., & Prasanna, V. (2019). Graphsaint: Graph sampling based inductive learning method. In *International conference on learning representations*.

Zhang, G., Wu, J., Yang, J., Beheshti, A., Xue, S., Zhou, C., & Sheng, Q. Z. (2021). Fraudre: Fraud detection dual-resistant to graph inconsistency and imbalance. In *2021 IEEE international conference on data mining (ICDM)* (pp. 867–876). IEEE.

Zhang, J., Yang, F., Lin, K., & Lai, Y. (2022). Hierarchical multi-modal fusion on dynamic heterogeneous graph for health insurance fraud detection. In *2022 IEEE international conference on multimedia and expo (ICME)* (pp. 1–6.) IEEE.

Zhong, Q., Liu, Y., Ao, X., Hu, B., Feng, J., Tang, J., & He, Q. (2020). Financial defaulter detection on online credit payment via multi-view attributed heterogeneous information network. In *Proceedings of the web conference 2020* (pp. 785–795).

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Authors and Affiliations

Lingfei Ren[1,2] · Ruimin Hu[1,2,5] · Yang Liu[3] · Dengshi Li[1,4] · Junhang Wu[1,2] · Yilong Zang[1,2] · Wenyi Hu[1,2]

✉ Ruimin Hu
hrm@whu.edu.cn

Lingfei Ren
renlingfei@whu.edu.cn

Yang Liu
liuyang17z@ict.ac.cn

Dengshi Li
reallds@jhun.edu.cn

Junhang Wu
wjh920925@whu.edu.cn

Yilong Zang
zangyl@whu.edu.cn

Wenyi Hu
wendyhu1028@gmail.com

[1] National Engineering Research Center for Multimedia Software, School of Computer Science, Wuhan University, Wuhan, China

[2] Hubei Key Laboratory of Multimedia and Network Communication Engineering, Wuhan University, Wuhan, China

[3] Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China

[4] School of Artificial Intelligence, Jianghan University, Wuhan, China

[5] School of Cyber Engineering, Xidian University, Xi'an, China