Pick and Choose: A GNN-based Imbalanced Learning Approach for Fraud Detection

Yang Liu[1,2], Xiang Ao*[1,2], Zidi Qin[1,2], Jianfeng Chi[3], Jinghua Feng[3], Hao Yang[3], Qing He[1,2,4]
[1]Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS),
Institute of Computing Technology, CAS, Beijing 100190, China
[2]University of Chinese Academy of Sciences, Beijing 100049, China
[3]Alibaba Group, Hangzhou, China
[4]Henan Institutes of Advanced Technology, Zhengzhou University, Zhengzhou 450052, China
{liuyang17z,aoxiang,qinzidi20s,heqing}@ict.ac.cn,{bianfu.cjf,jinghua.fengjh,youhiroshi.yangh}@alibaba-inc.com

ABSTRACT

Graph-based fraud detection approaches have escalated lots of attention recently due to the abundant relational information of graph-structured data, which may be beneficial for the detection of fraudsters. However, the GNN-based algorithms could fare poorly when the label distribution of nodes is heavily skewed, and it is common in sensitive areas such as financial fraud, etc. To remedy the class imbalance problem of graph-based fraud detection, we propose a Pick and Choose Graph Neural Network (PC-GNN for short) for imbalanced supervised learning on graphs. First, nodes and edges are picked with a devised label-balanced sampler to construct sub-graphs for mini-batch training. Next, for each node in the sub-graph, the neighbor candidates are chosen by a proposed neighborhood sampler. Finally, information from the selected neighbors and different relations are aggregated to obtain the final representation of a target node. Experiments on both benchmark and real-world graph-based fraud detection tasks demonstrate that PC-GNN apparently outperforms state-of-the-art baselines.

CCS CONCEPTS

• **Computing methodologies → Neural networks**; • **Security and privacy** → *Software and application security.*

KEYWORDS

class imbalance, graph neural network, fraud detection

ACM Reference Format:
Yang Liu, Xiang Ao, Zidi Qin, Jianfeng Chi, Jinghua Feng, Hao Yang, Qing He. 2021. Pick and Choose: A GNN-based Imbalanced Learning Approach for Fraud Detection. In *WWW '21: 30th The Web Conference, April 19–23, 2021, Virtual Event, Ljubljana, Slovenia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3442381.3449989

*Corresponding Author.

1 INTRODUCTION

Fraud detection is a vital task, with numerous high-impact applications in areas such as security [29], finance [22, 39, 49], health care [17], and review management [10, 26, 34]. Though numerous techniques have been developed in past years for detecting fraudsters in collections of multi-dimensional points, with the graph data becoming ubiquitous, graph-based fraud detection [1, 3, 33] has been focused on recently. At its heart, the basic assumption of graph-based fraud detection is that users, as well as fraudsters, have rich behavioral interactions when they purchase products or post reviews, and such interactions can be represented as graph-like data which renders effective multifaceted information for fraud detection.

However, the number of fraudsters can be far less than that of benign ones in the task of fraud detection. For example, in the real-world review dataset YelpChi [34] from Yelp.com, 14.5% of the reviews are spams while others are regarded as recommended reviews. In a real-world financial dataset [49] from Alibaba Group, only 0.5% of the users are defaulters who are not able to repay the credit debts borrowed from the financial platforms. As a result, the graph-based algorithms for fraud detection often suffer from the class imbalance problem and perform poorly especially for the minority but more important class, i.e., the fraudsters.

Recent years have witnessed research efforts devoting to resolving the class imbalance problem in conventional feature-based supervised learning settings, and they mainly fall into two directions, namely *re-sampling* and *re-weighting* methods. Re-sampling methods balance the number of examples by over-sampling the minority class [5, 24], or under-sampling the majority class [32]. Re-weighting methods assign different weights to different classes or even different samples by cost-sensitive adjustments [4, 9, 19, 21] or meta-learning based methods [14, 35, 37].

While the class imbalanced supervised learning in traditional feature space is well-studied, graph neural network algorithms that exclusively study the class imbalance problem are underexplored. DR-GCN [36] is a pioneer work to tackle the class imbalance problem on graphs. This method proposes a class-conditioned adversarial regularizer and a latent distribution alignment regularizer but cannot scale to large graphs.

We emphasize three major challenges from two sides in designing class imbalanced graph neural networks for graph-based fraud detection.

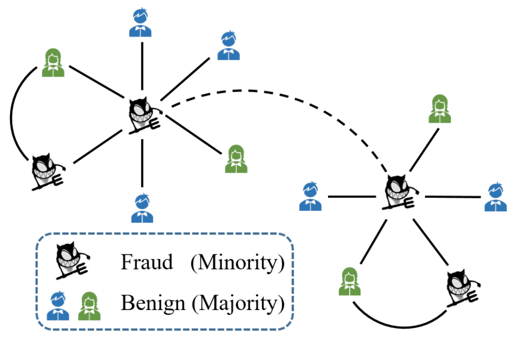


Figure 1: Illustration of the challenges in graph-based fraud detection. For the left fraud center node, only one of the six neighbors belongs to the same fraud class due to class imbalance. Thus the messages from the fraud neighbor are easily obscured during the message passing process. Besides, the neighborhood of the right fraud center node is similar to that of the left fraud center node, but they are not connected.

From the application side, the fraudsters may fabricate noisy information to make them hard to be identified, such as camouflage [10]. The first challenge caused by this is the redundant link information. For example, for spammers, they would employ benign accounts to post their spam reviews so that there will be many links between the spam review and the benign users, and the spammers would conceal themselves among benign users. Many feature-based or label-based similarities would fail to identify this kind of noisy neighbors since a fraudster could be in a close Euclidean distance with a benign one but their labels would be different. The second challenge derived from camouflage is the lack of necessary link information for fraudsters. For example, in financial settings, fraudsters would avoid trading with each other in order not to be detected together. As Figure 1 shows, the left fraud center node have similar transaction patterns with the right center node thus the feature of the right fraud node would be crucial for identifying the left fraud node. However, there is no link between the two nodes, which will decline the performance of GNN-based methods.

From the algorithm side, the challenge comes from the message aggregation of GNNs[11, 16, 38, 43, 44], which may lead to the result that the features of the minority class are diluted. Recall that the key design of graph neural networks lies in the neighborhood aggregation, but in the imbalanced settings, most neighbors of a center node might belong to the majority class. For example, as shown in Figure 1, only one of the six neighbors belongs to the same fraud class as the left center node. As a result, the features of the fraud neighbor could be easily overlooked and the prediction would be easily dominated by the majority benign ones.

To address the above challenges, in this paper, we propose a GNN-based imbalanced learning approach for graph-based fraud detection. For the algorithm side's challenge, we design a label-balanced sampler to **pick** nodes and edges to train. The probability assigned for each node is inversely proportional to its label frequency so that nodes of the minority class are more likely to be picked. As a result, the induced sub-graph of the picked nodes would have a balanced label distribution. For the challenges of the application side, we propose a neighborhood sampler to **choose** neighbors with a learnable parameterized distance function. For the fraud target node, the redundant links could be filtered by choosing neighbors that are far from the target measured by the distance and removing them from the neighbor set. And the necessary links which are beneficial for fraud prediction would be created by choosing similar nodes of the fraud class and regarding them as neighbors.

We integrate the above two stages of graph sampling and neighbor selection into general GNN frameworks and name our model as Pick and Choose Graph Neural Network (PC-GNN). Our contribution could be listed as follows.

- We formulate the graph-based fraud detection problem as an imbalanced node classification task and propose a GNN-based imbalanced learning approach to resolve the class imbalance problem on graphs.
- We design a label-balanced sampler to pick nodes and edges for sub-graph training and a neighborhood sampler to choose neighbors for over-sampling the neighborhood of the minority class and under-sampling the neighborhood of the majority class.
- Extensive experiments are conducted on two public benchmark datasets and two real-world datasets to verify the effectiveness of the proposed framework.

The remainder of the this paper is organized as follows. Section 2 introduces definitions and the problem statement of this paper. Section 3 details the proposed PC-GNN framework, and Section 4 illustrates the experiments. Section 5 surveys the related researches in the literature and Section 6 concludes the paper.

2 DEFINITION AND PROBLEM STATEMENT

2.1 Definition

Definition 2.1 (Imbalanced Ratio). Given a set of labels C, C_1 and C_2 denote two classes in C. The imbalance ratio of C_1 and C_2 is defined as $\mathrm{IR} = \frac{|C_1|}{|C_2|}$. Therefore, IR lies in the range $[0, +\infty)$. If $\mathrm{IR} > 1$, C_1 is called the majority class, and C_2 is called the minority class. Specially, if $\mathrm{IR} = 1$, C is balanced.

Definition 2.2 (Multi-Relation Imbalanced Graph). Given graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A}, \mathcal{X}, C)$, $\mathcal{V} = \{v_1, \ldots, v_N\}$ is a set of nodes, $\mathcal{E} = \{\mathcal{E}_1, \ldots, \mathcal{E}_R\}$ is the edge set of R relations, $\mathcal{A} = \{A_1, \ldots, A_R\}$ is a set of corresponding adjacency matrices of R relations. For each node $v_i \in \mathcal{V}$, $\mathbf{x}_i \in \mathcal{X}$ is a d-dimension feature vector and $c_i \in C$ is a scalar label, $i = 1, \ldots, N$. $\mathcal{X}$ and C are the set of node features and labels, respectively. If the imbalanced ratio of two classes in C is far larger than 1, we call $\mathcal{G}$ as a multi-relation imbalanced graph.

2.2 Problem Statement

Definition 2.3 (Graph-based Fraud Detection). The graph-based fraud detection problem is defined on the multi-relation imbalanced graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A}, \mathcal{X}, C)$, which has been formulated in Definition 2.2. Each node in $\mathcal{V}$ has been labeled either *fraud* or *benign* in C. Graph-based fraud detection is to find the fraud nodes that significantly differ from the other benign nodes on the multi-relation imbalanced graph $\mathcal{G}$, which can be formulated as an imbalanced node classification problem on $\mathcal{G}$.

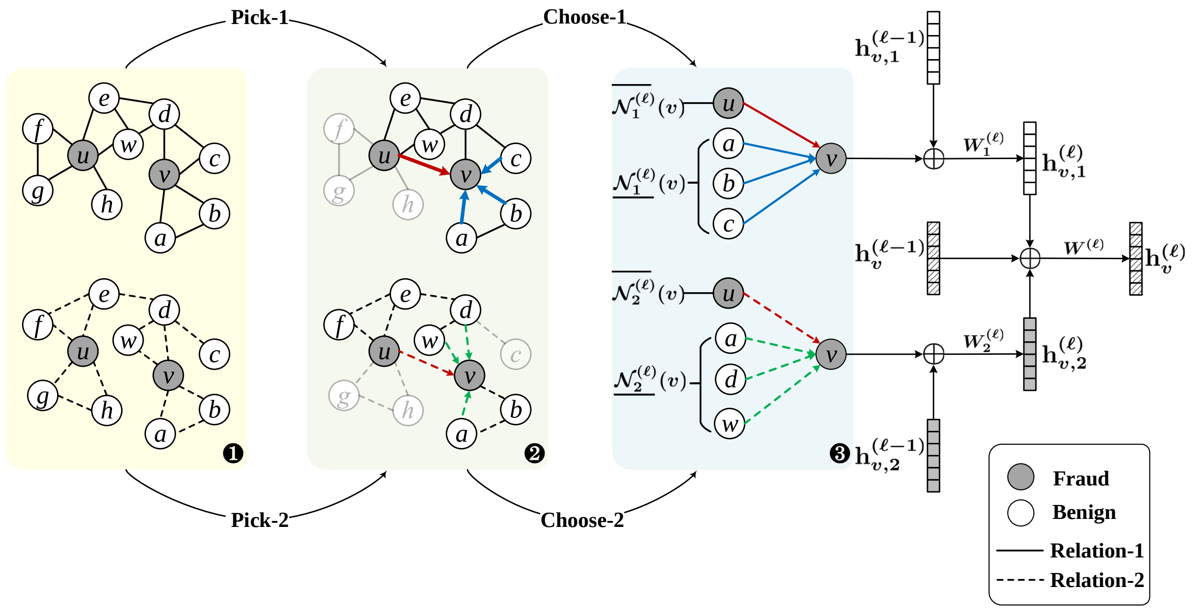

Pick-1
Choose-1
Pick-2
Choose-2
$\overline{\mathcal{N}_1^{(\ell)}}(v)$
$\underline{\mathcal{N}_1^{(\ell)}}(v)$
$\overline{\mathcal{N}_2^{(\ell)}}(v)$
$\underline{\mathcal{N}_2^{(\ell)}}(v)$
$\mathbf{h}_{v,1}^{(\ell-1)}$
$\mathbf{h}_{v,1}^{(\ell)}$
$W_1^{(\ell)}$
$\mathbf{h}_{v}^{(\ell-1)}$
$W^{(\ell)}$
$\mathbf{h}_{v}^{(\ell)}$
$W_2^{(\ell)}$
$\mathbf{h}_{v,2}^{(\ell)}$
$\mathbf{h}_{v,2}^{(\ell-1)}$
Fraud
Benign
Relation-1
Relation-2

Figure 2: The figure demonstrates the ℓ-th layer of the proposed PC-GNN framework on an example graph. ❶ shows an example graph containing 11 nodes. The solid and dashed lines represent two kinds of relations among these nodes. Nodes in gray are fraud ones and white are benign. From ❶ to ❷, a subset of nodes and edges are picked with the label-balanced sampler to construct sub-graphs for mini-batch training. ❷ illustrates the sub-graph induced by picked nodes, where nodes and edges that are not sampled are blurred. From ❷ to ❸, the neighbors are chosen by the neighborhood sampler. For fraud node v in ❷, the neighborhood is over-sampled by $\overline{\mathcal{N}_1^{(\ell)}}(v) = \{u\}$ which is similar to but not directly connected to v. Also, the original neighbor set of v is under-sampled by $\underline{\mathcal{N}_1^{(\ell)}}(v) = \{a, b, c\}$, where dissimilar nodes under the learned distance function are removed. Under different relations, as ❸ demonstrates, the chosen neighbors for v may be different, e.g. $\underline{\mathcal{N}_2^{(\ell)}}(v) = \{a, d, w\}$ differs from $\underline{\mathcal{N}_1^{(\ell)}}(v)$. Finally, all the neighbor information are aggregated and embeddings from different relations are concatenated to get the final representation of v at layer ℓ, denoted as $\mathrm{h}_v^{(\ell)}$ in the figure.

3 METHODOLOGY

In this section, we present the proposed PC-GNN framework. Firstly, we give an overview of the whole framework. Then, we detail the pick process and choose process in Section 3.2 and 3.3 respectively. Next, we explain how to aggregate information from different neighbors and relations in Section 3.4.

3.1 Overview

We illustrate the pipeline of the proposed framework on an example graph in Figure 2. To obtain the representation of the target entity, there are mainly three steps: pick, choose, and aggregate.

In the pick step, we design a label-balanced sampler to pick nodes and edges for sub-graph training. Next, in the choose step, we devise a neighborhood sampler to over-sample neighbors for the minority class and under-sample neighbors for the majority class. Finally, in the aggregation step, we aggregate the information from sampled neighbors and different relations.

3.2 Pick: Label-balanced Sampler

We devise a label-balanced graph sampler to pick nodes and edges for sub-graph construction. The key idea lies in incorporating label distribution information into the sampling process. For node sampler, those of the minority class have a higher sampling probability than the majority class.

Formally, $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A}, \mathcal{X}, C)$ is a multi-relation imbalanced graph, $A = \sum_{r=1}^{R} A_r$ is the sum of adjacency matrices of all relations, $\hat{A} = D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$ is the normalized adjacency matrix where D is a diagonal matrix with degree of each node as its element. For node $v \in \mathcal{V}$, its sampling probability is defined as Eq. (1).

$$P(v) \propto \frac{\|\hat{A}(:, v)\|^2}{\mathrm{LF}(C(v))} \tag{1}$$

where $\hat{A}(:, v)$ is the column of v in the normalized adjacency matrix $\hat{A}$, and $\mathrm{LF}(C(v))$ denotes the label frequency of class $C(v)$. The set of picked nodes is marked as $\mathcal{V}_p$, and $\mathcal{G}_p = (\mathcal{V}_p, \mathcal{E}_p, \mathcal{A}_p, \mathcal{X}_p, C_p)$ is the sub-graph induced by $\mathcal{V}_p$ and their one-hop neighbors.

3.3 Choose: Neighborhood Sampler

After the pick step, the following steps are conducted on the induced sub-graph $\mathcal{G}_p$. For symbolic clarity, we omit the subscript p in the following sections, i.e. we denote the adjacency matrices in $\mathcal{A}_p$ as $\{A_r\}_{r=1}^R$ and edges in $\mathcal{E}_p$ as $\{\mathcal{E}_r\}_{r=1}^R$. The neighbors of node v under each relation $\mathcal{E}_r$ are collected in the set $\mathcal{N}_r(v)$, as Eq. (2) shows.

$$\mathcal{N}_r(v) = \{u \in \mathcal{V} | A_r(v,u) > 0\} \tag{2}$$

As discussed in Section 1, it is not appropriate to adopt such neighbor definition as Eq. (2) in the imbalance graph since $\mathcal{N}_r(v)$ may contain disguised neighbors or lack necessary nodes which are crucial for prediction. In order to alleviate this problem, we should under-sample the neighborhood of the majority class to filter those noisy neighbors and over-sample the neighborhood of the minority class to add useful edges.

Practically, we add a constraint in the definition of the neighborhood of the majority class to filter those neighbors that are far from the target node under a certain distance function. The under-sampled neighbor set of node v is denoted by $\underline{\mathcal{N}_r}(v)$, as Eq. (3) shows, and it is obvious that $\underline{\mathcal{N}_r}(v) \subset \mathcal{N}_r(v)$.

$$\underline{\mathcal{N}_r}(v) = \{u \in \mathcal{V} | A_r(v,u) > 0 \text{ and } \mathcal{D}(v,u) < \rho\} \tag{3}$$

3.3.1 Distance Function. The distance function $\mathcal{D}(\cdot,\cdot)$ depends on the particular metric in the latent space. The widely-used distance function in latent space is the Euclidean distance of the features, i.e. $\mathcal{D}(v,u) = \|\mathbf{x}_v - \mathbf{x}_u\|$, where $\mathbf{x}_v \in \mathbb{R}^d$ denotes the features of node v. However, this distance function is not flexible in fraud detection since it does not consider the label information. Therefore, inspired by LAGCN[6], we adopt a parameterized distance function which combines the latent embeddings and the true label information and is defined as follows.

$$\mathcal{D}_r^{(\ell)}(v,u) = \left\| \mathrm{D}_r^{(\ell)}\left(\mathbf{h}_{v,r}^{(\ell)}\right) - \mathrm{D}_r^{(\ell)}\left(\mathbf{h}_{u,r}^{(\ell)}\right) \right\|_1 \tag{4}$$

where $\mathrm{D}_r^{(\ell)}\left(\mathbf{h}_{v,r}^{(\ell)}\right) = \sigma\left(\mathbf{U}_r^{(\ell)}\mathbf{h}_{v,r}^{(\ell)}\right)$ is a fully-connected layer predicting the probability of fraud based on the learnt embedding $\mathbf{h}_{v,r}^{(\ell)} \in \mathbb{R}^{d_\ell}$ of node v at layer ℓ under relation $\mathcal{E}_r$ and $\mathbf{U}_r^{(\ell)} \in \mathbb{R}^{1\times d_\ell}$ is the weight matrix of the distance function. The distance is defined as the difference of predicted probabilities of v and u.

3.3.2 Neighborhood Sampling. Therefore, the under-sampled neighborhood is re-written as Eq. (5).

$$\underline{\mathcal{N}_r^{(\ell)}}(v) = \left\{u \in \mathcal{V} | A_r(v,u) > 0 \text{ and } \mathcal{D}_r^{(\ell)}(v,u) < \rho_-\right\} \tag{5}$$

Besides, the neighborhood of v (belonging to the minority class) could be over-sampled by nodes which are far from v on the graph but have a certain similarity with v, which could be denoted as $\overline{\mathcal{N}_r^{(\ell)}}(v)$ and formulated as Eq. (6).

$$\overline{\mathcal{N}_r^{(\ell)}}(v) = \left\{u \in \mathcal{V} | C(u) = C(v) \text{ and } \mathcal{D}_r^{(\ell)}(v,u) < \rho_+\right\} \tag{6}$$

To sum up, for the target node v of the majority class, the neighborhood of v is under-sampled by $\mathcal{N}_r^{(\ell)}(v) = \underline{\mathcal{N}_r^{(\ell)}}(v)$; for the target node v of the minority class, the neighborhood of v is over-sampled by $\mathcal{N}_r^{(\ell)}(v) = \underline{\mathcal{N}_r^{(\ell)}}(v) \cup \overline{\mathcal{N}_r^{(\ell)}}(v)$.

3.3.3 Learning. The neighborhood sampler is learnable due to the parameterization of the distance function. The parameters of the distance function include the weights of $\mathrm{D}_r^{(\ell)}$, i.e. $\mathbf{U}_r^{(\ell)}$, which is optimized with cross-entropy loss as Eq. (7).

$$\begin{gathered}\mathcal{L}_{\text{dist}} = -\sum_{\ell=1}^{L}\sum_{r=1}^{R}\sum_{v\in\mathcal{V}}\left[y_v \log p_{v,r}^{(\ell)} + (1-y_v)\log\left(1-p_{v,r}^{(\ell)}\right)\right] \\ p_{v,r}^{(\ell)} = \mathrm{D}_r^{(\ell)}\left(\mathbf{h}_{v,r}^{(\ell)}\right)\end{gathered} \tag{7}$$

Algorithm 1: PC-GNN: Pick and Choose Graph Neural Network

```
Input: $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A}, \mathcal{X}, C)$: A multi-relation imbalanced
       graph, $\mathcal{V}_{\text{train}}$: Set of training nodes, $N_p$: Number of
       picked nodes in each epoch, $N_{\text{epoch}}$: Number of total
       training epochs, $N_{\text{batch}}$: Number of training batch
       size, $L$: Number of layers, $d_\ell$: Dimension of $\ell$-th
       layer, $\ell = 1, \ldots, L$.
Output: The vector representations for each node in $\mathcal{V}$.
1  Initialization $\mathbf{h}_v^{(0)} \leftarrow \mathbf{x}_v$, $P(v) \propto \frac{\|\hat{A}(:,v)\|^2}{\mathrm{LF}(C(v))}$, $v \in \mathcal{V}_{\text{train}}$;
2  for $e = 1, \ldots, N_{\text{epoch}}$ do
3      Pick $N_p$ nodes according to the probability defined in
         Eq. (1) to obtain $\mathcal{V}_p$;
4      Decide the number of training batches $B = \lceil \frac{|\mathcal{V}_p|}{N_{\text{batch}}} \rceil$;
5      for $b = 1, \ldots, B$ do
6          Gather nodes of batch $b$ along with edges between
             them to construct sub-graph $\mathcal{G}_b = (\mathcal{V}_b, \mathcal{E}_b)$;
7          for $\ell = 1, \ldots, L$ do
8              for $r = 1, \ldots, R$ do
9                  For each node $v \in \mathcal{V}_b$, its neighbors are
                     under-sampled as Eq. (5). If $v$ belongs to
                     the minority class, the neighborhood is
                     further over-sampled by Eq. (6);
10                 Update $\mathbf{h}_{v,r}^{(\ell)}$ w.r.t. Eq. (8), $v \in \mathcal{V}_b$
11             Update $\mathbf{h}_v^{(\ell)}$ w.r.t. Eq. (9), $v \in \mathcal{V}_b$;
12 return $\mathbf{h}_v^{(L)}$, $v \in \mathcal{V}$.
```

3.4 Aggregate: Message Passing Architecture

After the choose step, $\mathcal{N}_r^{(\ell)}(\cdot)$ collects the over-sampled neighborhood of the minority class or the under-sampled neighborhood of the majority class at layer ℓ under relation $\mathcal{E}_r$. Message passing based graph neural network is designed to aggregate information from all the neighbors and relations. Let $\mathbf{h}_{v,r}^{(\ell)} \in \mathbb{R}^{d_\ell}$ denote the representation of node v at layer ℓ under relation $\mathcal{E}_r$, where $v \in \mathcal{V}$, $r = 1, \ldots, R$, $\ell = 1, \ldots, L$ and L is the number of layers.

The aggregate step is further divided into two steps. First, under each relation, aggregate all the information from selected neighbors like Eq. (8) shows, where $\text{AGG}_r^{(\ell)}$ is the mean aggregator function at layer ℓ under relation $\mathcal{E}_r$, $\oplus$ denotes the concat operation, and $W_r^{(\ell)} \in \mathbb{R}^{d_\ell \times 2d_{\ell-1}}$ is the weight matrix.

$$\mathbf{h}_{v,r}^{(\ell)} = \text{ReLU}\left(W_r^{(\ell)}\left(\mathbf{h}_{v,r}^{(\ell-1)} \oplus \text{AGG}_r^{(\ell)}\{\mathbf{h}_{u,r}^{(\ell-1)}, u \in \mathcal{N}_r^{(\ell)}(v)\}\right)\right) \quad (8)$$

Then, we need to combine each $\mathbf{h}_{v,r}^{(\ell)}$ with the representation from the previous layer $\mathbf{h}_v^{(\ell-1)}$ to obtain $\mathbf{h}_v^{(\ell)}$ in the ℓ-th layer, which is illustrated as Eq. (9) and $W^{(\ell)} \in \mathbb{R}^{d_\ell \times (d_{\ell-1} + R \cdot d_\ell)}$ is the weight matrix.

$$\mathbf{h}_v^{(\ell)} = \text{ReLU}\left(W^{(\ell)}\left(\mathbf{h}_v^{(\ell-1)} \oplus \mathbf{h}_{v,1}^{(\ell)} \oplus \cdots \oplus \mathbf{h}_{v,R}^{(\ell)}\right)\right) \quad (9)$$

3.5 Training

Following the aggregation step, an MLP classifier is trained together with graph neural networks to minimize the cross-entropy loss.

$$\begin{aligned} \mathcal{L}_{\text{gnn}} &= -\sum_{v \in \mathcal{V}} \left[y_v \log p_v + (1 - y_v) \log(1 - p_v)\right] \\ p_v &= \text{MLP}\left(\mathbf{h}_v^{(L)}\right) \end{aligned} \quad (10)$$

And the overall loss function is formulated as (11), where α is a balanced parameter.

$$\mathcal{L} = \mathcal{L}_{\text{gnn}} + \alpha \mathcal{L}_{\text{dist}} \quad (11)$$

The overall training algorithm is summarized in Algorithm 1. Given a multi-relation imbalanced graph $\mathcal{G}$ and the training node set $\mathcal{V}_{\text{train}}$, we first pick nodes from $\mathcal{V}_{\text{train}}$ according to the sampling probability in Eq. (1) for training (Line 3). These nodes are split into mini-batches with the size of N_{batch} (Line 6). For the node in each sub-graph of each batch, its neighbors are over-sampled or under-sampled (Line 9) according to its label frequency. Then messages from chosen neighbors are aggregated (Line 10) and representations of different relations are concatenated (Line 11).

4 EXPERIMENTS

In this section, we investigate the effectiveness of the proposed PC-GNN model on two kinds of graph-based fraud detection tasks, namely opinion fraud detection and financial fraud detection, with the aim of answering the following research questions.

- **RQ1**: Does PC-GNN outperform the state-of-the-art methods for graph-based anomaly detection?
- **RQ2**: How do the key components benefit the prediction?
- **RQ3**: What is the performance with respect to different training parameters?
- **RQ4**: If the proposed modules are applied to other GNN models, will it bring performance improvement?

4.1 Experimental Setup

4.1.1 Dataset. Opinion fraud detection aims to find the spam reviews from online platform like Yelp.com and Amazon.com. We adopt two datasets for this task. The **YelpChi** dataset [34] collects hotel and restaurant reviews on Yelp. The nodes in the graph of YelpChi dataset are reviews with 100-dimension features and have three relations: 1) R-U-R denotes the reviews posted by the same user; 2) R-S-R denotes the reviews under the same product with the same star rating; 3) R-T-R denotes the reviews under the same product posted in the same month. The **Amazon** dataset[28] includes product reviews under the Musical Instrument category. The nodes in the graph of Amazon dataset are users with 100-dimension features and also contain three relations: 1) U-P-U connects users reviewing at least one same product; 2) U-S-U connects users having at least one same star rating within one week; 3) U-V-U connects users with top-5% mutual review TF-IDF similarities. The statistics of these two datasets are shown in Table 1.

Table 1: Statistics of Review Datasets

Dataset	#Node	#Edge	IR	Relations	#Relations
YelpChi	45,954	3,846,979	5.9	R-U-R	49,315
				R-S-R	3,402,743
				R-T-R	573,616
Amazon	11,944	4,398,392	13.5	U-P-U	175,608
				U-S-U	3,566,479
				U-V-U	1,036,737

Financial fraud detection is to find the default users from the user behavioral data on financial service platforms. We collect two real-world datasets from an online credit payment service provided by Alibaba Group, namely **M7** and **M9**. M7 collects users from from 2018/07/01 to 2018/07/31 and M9 collects users from from 2018/09/01 to 2018/09/30. The feature dimension of users is 908. There are four kinds of relations among these users: 1) U-T-U means that one user trades to another; 2) U-D-U means that two users log in the same device this month; 3) U-F-U means that the users have fund transfer records this month; 4) U-S-U means that the users have social relationship like family or workmate. The statistics of these two datasets are shown in Table 2.

Table 2: Statistics of Financial Datasets

Dataset	#Node	#Edge	IR	Relations	#Relations
M7	188,673	2,239,344	118.4	U-T-U	2,179,770
				U-D-U	28,630
				U-F-U	24,724
				U-S-U	6,220
M9	253,221	5,568,580	141.5	U-T-U	5,483,056
				U-D-U	40,354
				U-F-U	36,644
				U-S-U	8,526

4.1.2 Compared Methods. We compare with several state-of-the-art graph neural network methods and their enhancements to verify the effectiveness of our proposed method in graph-based fraud detection.

- **GCN** [16]: graph convolution network achieved by a localized first-order approximation of spectral graph convolutions.

Table 3: Performance comparison on YelpChi and Amazon for opinion fraud detection

Method	Dataset	YelpChi			Amazon		
	Metric	F1-macro	AUC	GMean	F1-macro	AUC	GMean
Baselines	GCN	0.5620±0.0067	0.5983±0.0049	0.4365±0.0262	0.6486±0.0694	0.8369±0.0125	0.5718±0.1951
	GAT	0.4879±0.0230	0.5715±0.0029	0.1659±0.0789	0.6464±0.0387	0.8102±0.0179	0.6675±0.1345
	DR-GCN	0.5523±0.0231	0.5921±0.0195	0.4038±0.0742	0.6488±0.0364	0.8295±0.0079	0.5357±0.1077
	GraphSAGE	0.4405±0.1066	0.5439±0.0025	0.2589±0.1864	0.6416±0.0079	0.7589±0.0046	0.5949±0.0349
	GraphSAINT	0.5960±0.0038	0.6999±0.0029	0.5908±0.0298	0.7626±0.0032	0.8701±0.0025	0.7963±0.0091
	GraphConsis	0.5870±0.0200	0.6983±0.0302	0.5857±0.0385	0.7512±0.0325	0.8741±0.0334	0.7677±0.0486
	CARE-GNN	0.6332±0.0094	0.7619±0.0292	0.6791±0.0359	0.8990±0.0073	0.9067±0.1115	0.8962±0.0018
Ablation	PC-GNN$_{\setminus P}$	0.5136±0.0147	0.7844±0.0013	0.2336±0.0356	**0.9158±0.0024**	0.9469±0.0018	0.8782±0.0068
	PC-GNN$_{\setminus C}$	**0.6634±0.0058**	0.7847±0.0021	0.6258±0.0378	0.8929±0.0171	0.9529±0.0035	0.9006±0.0045
Ours	PC-GNN	0.6300±0.0230	**0.7987±0.0014**	**0.7160±0.0130**	0.8956±0.0077	**0.9586±0.0014**	**0.9030±0.0044**

- **GAT** [38]: graph attention network that employs attention mechanism for neighbor aggregation.
- **DR-GCN** [36]: a dual-regularized graph convolutional network to handle multi-class imbalanced graph representation learning.
- **GraphSAGE** [11]: an inductive GNN model based on a fixed sample number of the neighbor nodes.
- **GraphSAINT** [47]: a scalable and efficient GNN model based on graph sampling.
- **GraphConsis** [26]: a heterogeneous graph neural network which tackles context inconsistency, feature inconsistency and relation inconsistency problem.
- **CARE-GNN** [10]: a camouflage-resistant graph neural network which enhances the GNN aggregation process with three unique modules against camouflages.
- **PC-GNN**: our proposed method. We also derive two variants of PC-GNN to comprehensively compare and analyze the performances of its each component. They are,
- **PC-GNN**$_{\setminus P}$: removing label-balanced sampler and following the original label distribution in sub-graph sampling.
- **PC-GNN**$_{\setminus C}$: removing neighborhood sampler and aggregating messages from all topological neighbors.

4.1.3 Experimental Settings. The parameters of PC-GNN are optimized with Adam[15] optimizer, and the learning rate is set to be 0.01. In PC-GNN, N_p is twice the size of the minority class, $N_{\text{epoch}} = 100$, $L = 1$, $d_1 = 64$, $\alpha = 2$, $N_{\text{batch}} = 1024$, ρ_- is determined by the top-50% distance value of each neighborhood, and ρ_+ is decided by the top-k distance of the minority class, where k equals half the average neighborhood size of the minority class. In GraphSAINT, the node size of sampled sub-graph is set to be 5000. In GraphSAGE, the neighborhood size is set to be 5. For all the compared methods, we report the average value and standard deviation of 10 runs. The train, valid, and test ratio are set to be 40%, 20%, and 40% respectively. The train-test split is based on the stratified sampling provided by Scikit-learn[31] to ensure that the imbalance ratio is consistent in both train and test set.

The original GCN, DR-GCN and GraphSAGE algorithms suffer from the imbalanced problem and produce few positive predictions. Thus we adjust the binary classification threshold of these three algorithms to achieve the best F1-macro and GMean scores for fair comparison, which is called threshold-moving strategy in the literature[8]. Specifically, for YelpChi and Amazon, the classification threshold is set to be 0.2. For financial datasets M7 and M9, the threshold is further cut down to 0.008.

4.1.4 Implementation. PC-GNN is implemented in Pytorch 1.6.0 [30] with Python 3.7 and all the experiments are run on Ubuntu 16.04.5 LTS server with 40 cores and 128GB memory. GCN, GraphSAGE and GAT are implemented based on DGL[41]. And DR-GCN is implemented by ourselves since the authors do not provide source code in their paper. GraphSAINT, GraphConsis and CARE-GNN are implemented using their provided source code.

4.1.5 Metrics. For class imbalance classification, the evaluation metrics should have no bias to any class[27]. Therefore, we use three widely adopted metrics to measure the performance of all the compared methods, namely **F1-macro**, **AUC** and **GMean**.

The first metric **F1-macro** is the unweighted mean of the F1-score of each class.

The second metric **AUC** is the area under the ROC Curve and is defined as:

$$\text{AUC} = \frac{\sum_{u \in \mathcal{U}^+} rank_u - \frac{|\mathcal{U}^+| \times (|\mathcal{U}^+|+1)}{2}}{|\mathcal{U}^+| \times |\mathcal{U}^-|}$$

Here, $\mathcal{U}^+$ and $\mathcal{U}^-$ denotes the minority and majority class set in the testing set, respectively. And $rank_u$ indicates the rank of node u via the score of prediction.

The third metric **GMean** calculates the geometric mean of True Positive Rate (TPR) and True Negative Rate (TNR).

$$\text{GMean} = \sqrt{\text{TPR} \cdot \text{TNR}} = \sqrt{\frac{\text{TP}}{\text{TP}+\text{FN}} \cdot \frac{\text{TN}}{\text{TN}+\text{FP}}}$$

The higher scores of these metrics indicate the higher performance of the approaches.

Table 4: Performance comparison on M7 and M9 for financial fraud detection

Method	Dataset	M7			M9		
	Metric	F1-macro	AUC	GMean	F1-macro	AUC	GMean
Baselines	GCN	0.3108±0.0256	0.6107±0.0041	0.5456±0.0159	0.3016±0.0574	0.5790±0.0040	0.5241±0.0422
	GAT	0.2746±0.0168	0.6083±0.0149	0.5016±0.0168	0.2698±0.0069	0.5647±0.0069	0.4354±0.0346
	DR-GCN	0.3070±0.0232	0.7195±0.0208	0.5647±0.0403	0.5055±0.0012	0.6637±0.0236	0.3106±0.0417
	GraphSAGE	0.5186±0.0030	0.6790±0.0029	0.1605±0.0132	0.5020±0.0026	0.6342±0.0040	0.0525±0.0362
	GraphSAINT	0.5149±0.0036	0.6915±0.0068	0.2547±0.0459	0.5018±0.0019	0.6587±0.0049	0.1864±0.0354
	GraphConsis	0.5236±0.0087	0.6826±0.0049	0.2734±0.0548	0.5124±0.0043	0.6743±0.0076	0.2302±0.0467
	CARE-GNN	0.5578±0.0015	0.7836±0.0020	0.3451±0.0098	0.5361±0.0035	0.7579±0.0060	0.2908±0.0294
Ablation	PC-GNN$_{\backslash P}$	0.4979±0.0000	0.7434±0.0042	0.0000±0.0000	0.4982±0.0000	0.6575±0.0069	0.0000±0.0000
	PC-GNN$_{\backslash C}$	0.5735±0.0017	0.8132±0.0031	0.4362±0.0254	0.5353±0.0028	0.7668±0.0038	0.3138±0.0387
Ours	PC-GNN	**0.5749±0.0044**	**0.8192±0.0032**	**0.6645±0.0422**	**0.5370±0.0021**	**0.7847±0.0019**	**0.5740±0.0391**

Table 5: F1 scores of each class in the ablation study for opinion fraud detection

Dataset	YelpChi		Amazon	
Metric	F1-fraud	F1-benign	F1-fraud	F1-benign
PC-GNN$_{\backslash P}$	0.1032±0.0291	**0.9240±0.0004**	**0.8463±0.0045**	**0.9853±0.0003**
PC-GNN$_{\backslash C}$	0.4271±0.0191	0.8997±0.0095	0.8070±0.0295	0.9788±0.0047
PC-GNN	**0.4344±0.0127**	0.8255±0.0335	0.8116±0.0133	0.9795±0.0021

4.2 Performance Comparison (RQ1)

To answer RQ1, we evaluate the performance of all the compared methods in the task of opinion fraud detection and financial fraud detection. The corresponding F1-macro, AUC, GMean scores are reported in Table 3 and 4 respectively. The compared methods can be further divided into three groups.

First, GraphConsis and CARE-GNN are two state-of-the-art methods for graph-based fraud detection, focusing on inconsistency and camouflage problems respectively. Different from them, our PC-GNN adopts a label-balanced sampler to pick nodes and a neighborhood sampler to choose neighbors for aggregation. And experimental results demonstrate that PC-GNN outperforms these two methods, with 3%~5% improvement in AUC and 0.7%~28% improvement in GMean. The F1-scores of PC-GNN and CARE-GNN are comparable and higher than that of GraphConsis. CARE-GNN performs better than GraphConsis in all these metrics which is consistent with the conclusion in the paper of CARE-GNN.

Second, GraphSAGE and GraphSAINT are two representative sampling-based methods, which employ node sampling and graph sampling respectively. However, these two algorithms do not explore label distribution when sampling, thus they perform worse than PC-GNN. GraphSAGE performs worse than GraphSAINT because GraphSAGE keeps a fixed size of the neighborhood, and for those nodes with a large size of the neighborhood, the corresponding information loss will decline its performance.

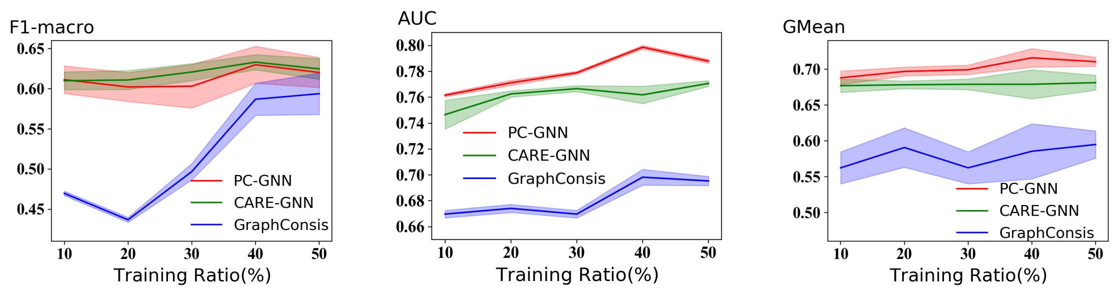
Third, GCN and GAT are traditional graph neural network methods, and DR-GCN is a dual-regularized GCN for multi-class imbalanced classification. In these methods, the minority class is not sufficiently trained due to its small number of samples, and therefore their performances are worst among the compared methods. GAT achieves the lowest scores among these three methods because the attention mechanism owns the most parameters, but the minority class lacks sufficient data to train a good model.

4.3 Ablation Study (RQ2)

To answer RQ2, we identify the two key steps of PC-GNN, i.e., pick and choose, and verify their effectiveness by removing each part, respectively. For the financial fraud detection, as illustrated in Table 4, PC-GNN gets the highest scores in the three metrics, which proves that the pick and choose step are both effective in handling the class imbalance in graph-based fraud detection. Due to the high (larger than 100) imbalance ratio of M7 and M9, PC-GNN$_{\backslash P}$ does not have positive predictions in these two datasets. Thus the GMean scores are zero and F1-macro scores have zero variance.

For the opinion fraud detection, as illustrated in Table 3, PC-GNN achieves the best performance in AUC and GMean compared with its two variants, which is consistent with the results of the financial fraud detection. However, PC-GNN gets a lower F1-macro score than one of the variants. To explore the reason, we further exhibit the F1-score of each class in the ablation study experiments in Table 5.

As Table 5 shows, the full model gets the best F1-score of the fraud class in Yelp while PC-GNN$_{\backslash P}$ and PC-GNN$_{\backslash C}$ perform better in the benign class. However, in the fraud detection task, the performance of the fraud class is more focused on than that of the



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

Figure 3: Sensitivity analysis with respect to different training ratios. The solid line represents the average score of 10 runs and the shadow indicates the standard deviation. Each method is marked in different colors.

Table 6: Performance of enhanced version of GCN and GraphSAGE on YelpChi and Amazon

Dataset	YelpChi			Amazon		
Metric	F1-macro	AUC	GMean	F1-macro	AUC	GMean
GCN	0.4645±0.0033	0.5983±0.0049	0.0483±0.0373	0.5297±0.0539	0.8369±0.0125	0.1904±0.1716
GCN(T)	**0.5620±0.0067**	0.5983±0.0049	0.4365±0.0262	0.6486±0.0694	0.8369±0.0125	0.5718±0.1951
GCN(P)	0.5540±0.0275	**0.6122±0.0643**	**0.5114±0.0107**	**0.7138±0.0086**	**0.8773±0.0027**	**0.7823±0.0374**
GraphSAGE	0.4608±0.0000	0.5439±0.0025	0.0000±0.0000	0.4751±0.0000	0.7589±0.0046	0.0000±0.0000
GraphSAGE(T)	0.4405±0.1066	0.5439±0.0025	0.2589±0.1864	**0.6416±0.0079**	0.7589±0.0046	0.5949±0.0349
GraphSAGE(P)	**0.6178±0.0286**	**0.7765±0.0025**	**0.6952±0.0176**	0.5831±0.0227	**0.7627±0.0097**	**0.6993±0.0081**

benign class, thus the full model still gets the best performance of predicting the fraud targets. Besides, we observe that in the Amazon dataset, PC-GNN$_{\setminus P}$ achieves the best F1-score in both classes. We argue the reason is that the graph size of Amazon is relatively small, and extracting sub-graphs from a small graph could cause loss of information. Therefore, when we remove the pick step in Amazon, the training is performed on the whole graph, and PC-GNN$_{\setminus P}$ gets the best results. We also conclude that PC-GNN is more effective on large-scale imbalanced graphs.

4.4 Sensitivity Analysis (RQ3)

To answer RQ3, we further evaluate the performance of PC-GNN with respect to the training ratio. We vary the percentage of training nodes from 10% to 50% and rerun our PC-GNN and two strong baselines, namely GraphConsis and CARE-GNN, for comparison. Figure 3 presents the scores of F1-macro, AUC, and GMean of the three compared methods when varying the training ratio. We can observe that, under each training ratio setting, PC-GNN always achieves the best performance in terms of AUC and GMean. The F1-macro scores of PC-GNN and CARE-GNN are comparable and surpass that of GraphConsis by a large margin. Therefore, we conclude that PC-GNN is robust to the training ratio and consistently outperforms GraphConsis and CARE-GNN.

4.5 Enhancement for other GNNs (RQ4)

To answer RQ4, we investigate how to equip the traditional GNN methods with the pick and choose modules to improve their performances in class-imbalanced graph-based fraud detection tasks. We argue that the pick step is independent of neighbor aggregation and is only used for sub-graph construction. Therefore it is possible that we modify the traditional GNN models to train in sub-graphs induced by the picked nodes for imbalanced fraud detection. And the choose step could influence the scheme of aggregation and cannot be easily adapted to other methods.

We choose two representative methods, namely GCN and GraphSAGE, and modify them into sub-graph training versions which are denoted by GCN(P) and GraphSAGE(P) respectively. The nodes in the sub-graphs are picked using the same probabilities proposed in this paper and the performances are illustrated in Table 6, where GCN(T) and GraphSAGE(T) represent the threshold-moving versions of GCN and GraphSAGE, of which the results are also reported in Table 3 and Table 4.

As we can conclude from Table 6, the modified models with the pick module consistently outperform the original algorithms and the threshold-moving algorithms. The threshold-moving strategy does not improve the performance in terms of AUC since AUC only depends on the prediction probability. But models with pick can improve values of all the three metrics, which proves that the proposed pick module is effective for imbalanced graph-based fraud detection.

5 RELATED WORK

This section introduces previous studies on imbalanced learning and graph-based fraud detection.

5.1 Imbalanced Learning

Existing algorithms for imbalanced learning can be divided into two groups, namely re-sampling and re-weighting approaches.

Re-sampling methods can be further divided into over-sampling and under-sampling algorithms. Random over-sampling (ROS) repeats samples of the minority class randomly among the dataset. SMOTE [5] is a representative interpolation-based over-sampling methods and also has lots of extensions [12, 13]. Generative methods like GLGAN [42], ADAAR [24] generate synthetic samples to augment the minority class for over-sampling. Random under-sampling (RUS) discards instances of majority class randomly from the dataset. Exploratory under-sampling [23] employs EasyEnsemble and BalanceCascade to overcome the deficiency that the majority class examples are ignored in under-sampling. Trainable under-sampling (TU) [32] incorporates evaluation metric optimization into the data sampling process to perform under-sampling. Recently, TRUST [7] proposes to learn how to learn an under-sampling strategy through meta-learning based reinforcement learning.

Re-weighting algorithms can be achieved by cost-sensitive approaches [4, 9, 19, 21] and meta-learning based methods [14, 35, 37]. Cost-sensitive re-weighting methods assign weights to each sample according to their individual properties. Focal loss [21] down-weights the well-classified examples. Dice loss [19] attaches similar importance to false positives and false negatives and is more immune to the data-imbalance issue. Label-distribution-aware loss [4] minimizes a margin-based generalization bound and could be applied with prior strategies for training. Learning to re-weight [35] learns to assign weights to training examples based on their gradient directions. Learning data manipulation [14] assigns the data sample importance dynamically.

Other approaches like transfer learning [45], metric learning [46] could also be used to resolve the class imbalance problem. Our work is different from these algorithms since we aim to remedy the special challenges of the class imbalance problem on graph structure which may render these algorithms ineffective and inapplicable.

5.2 Graph-based Fraud Detection

The graph-like data in fraud detection tasks are generally users' various behaviors and usually contain different relations. Therefore, heterogeneous graph-based algorithms are widely used for graph-based fraud detection, and we summarize existing works from two categories, i.e., financial fraud detection [2, 20, 25, 39, 48, 49] and opinion fraud detection [10, 18, 26, 40] according to the recent popularity. Other related researches could be referred to [33].

Financial fraud detection is to find malicious accounts, default users, and fraud transactions based on the behavioral data from the financial platforms. GEM [25] adaptively learns discriminative embeddings from heterogeneous account-device graphs for malicious accounts detection. Semi-GNN [39] is a semi-supervised GNN model with hierarchical attention mechanism for explainable fraud prediction. GAL [48] is a graph anomaly loss function that trains GNNs for anomaly-detectable node representations. MAHINDER [49] explores meta-paths on multi-view attributed heterogeneous information network for default user detection.

Opinion fraud detection, also known as spam review detection, is to detect the spam reviews that mislead customers or spammers who post spam reviews. FdGars [40] is a graph convolutional network approach for fraudster detection in online app review system. GAS [18] integrates both heterogeneous and homogeneous graphs to capture the local context and global context of a comment. GraphConsis [26] investigates the context, feature and relation inconsistency problem in graph-based fraud detection. CARE [10] enhances the GNN aggregation process against camouflage for opinion fraud detection.

Most of the mentioned approaches adopt the simple random over-sampling method to address the class imbalance problem on graphs. Different from them, we solve the problem by a novel pick and choose graph neural network framework.

6 CONCLUSION AND FUTURE WORK

In this work, we propose a GNN-based imbalanced learning method named PC-GNN to solve the class imbalance problem in graph-based fraud detection. The overall framework could be decoupled into three steps: Pick, Choose, and Aggregate. The center nodes are picked with a label-balanced sampler to construct a balanced sub-graph for mini-batch training. Under a parameterized distance function, the neighborhood of the minority class is over-sampled and that of the majority class is under-sampled. Messages from selected neighbors and different relations are aggregated to obtain the final representations of the target. Experiments on two benchmark opinion fraud datasets and two real-world financial fraud datasets demonstrate the effectiveness of the proposed framework. For future work, besides label-aware graph sampling methods, learning a new graph structure for the imbalanced graph could be a promising direction for graph-based fraud detection.

ACKNOWLEDGMENTS

The research work is supported by Alibaba Group through Alibaba Innovative Research Program. This work is also supported by the National Key Research and Development Program of China under Grant No. 2017YFB1002104, the National Natural Science Foundation of China under Grant No. 92046003, 61976204, U1811461, the Project of Youth Innovation Promotion Association CAS and Beijing Nova Program Z201100006820062. We would like to thank the anonymous reviewers for their valuable comments, and Mengda Huang, Linfeng Dong, Kuan Li for their insightful discussions.

REFERENCES

[1] Leman Akoglu, Hanghang Tong, and Danai Koutra. 2015. Graph Based Anomaly Detection and Description: A Survey. *Data Min. Knowl. Discov.* (2015).
[2] Xiang Ao, Yang Liu, Zidi Qin, Yi Sun, and Qing He. 2021. Temporal high-order proximity aware behavior analysis on Ethereum. *World Wide Web* (2021), 1–21.
[3] Paul Boniol, Themis Palpanas, Mohammed Meftah, and Emmanuel Remy. 2020. GraphAn: Graph-Based Subsequence Anomaly Detection. *VLDB Endowment* 13, 12 (2020), 2941–2944.
[4] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. 2019. Learning imbalanced datasets with label-distribution-aware margin loss. In *NeurIPS*.
[5] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. 2002. SMOTE: synthetic minority over-sampling technique. *JAIR* (2002).

[6] Hao Chen, Lu Wang, Senzhang Wang, Dijun Luo, Wenbing Huang, and Zhoujun Li. 2020. Label Aware Graph Convolutional Network–Not All Edges Deserve Your Attention. In *CIKM*.

[7] Jianfeng Chi, Guanxiong Zeng, Qiwei Zhong, Ting Liang, Jinghua Feng, Xiang Ao, and Jiayu Tang. 2020. Learning to Undersampling for Class Imbalanced Credit Risk Forecasting. In *ICDM*.

[8] Guillem Collell, Drazen Prelec, and Kaustubh R Patil. 2018. A simple plug-in bagging ensemble based on threshold-moving for classifying binary and multiclass imbalanced data. *Neurocomputing* 275 (2018), 330–340.

[9] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. 2019. Class-balanced loss based on effective number of samples. In *CVPR*.

[10] Yingtong Dou, Zhiwei Liu, Li Sun, Yutong Deng, Hao Peng, and Philip S Yu. 2020. Enhancing Graph Neural Network-based Fraud Detectors against Camouflaged Fraudsters. In *CIKM*.

[11] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *NeurIPS*.

[12] Hui Han, Wenyuan Wang, and Binghuan Mao. 2005. Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning. In *International Conference on Intelligent Computing*.

[13] Haibo He, Yang Bai, Edwardo A Garcia, and Shutao Li. 2008. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In *IJCNN*.

[14] Zhiting Hu, Bowen Tan, Ruslan Salakhutdinov, Tom M. Mitchell, and Eric P. Xing. 2019. Learning Data Manipulation for Augmentation and Weighting. In *NeurIPS 2019*. 15738–15749.

[15] Diederik P Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.

[16] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*.

[17] Mohit Kumar, Rayid Ghani, and Zhu-Song Mei. 2010. Data Mining to Predict and Prevent Errors in Health Insurance Claims Processing. In *KDD*.

[18] Ao Li, Zhou Qin, Runshi Liu, Yiqun Yang, and Dong Li. 2019. Spam Review Detection with Graph Convolutional Networks. In *CIKM*.

[19] Xiaoya Li, Xiaofei Sun, Yuxian Meng, Junjun Liang, Fei Wu, and Jiwei Li. 2020. Dice Loss for Data-imbalanced NLP Tasks. In *ACL*.

[20] Ting Liang, Guanxiong Zeng, Qiwei Zhong, Jianfeng Chi, Jinghua Feng, Xiang Ao, and Jiayu Tang. 2021. Credit Risk and Limits Forecasting in E-Commerce Consumer Lending Service via Multi-view-aware Mixture-of-experts Nets. In *WSDM*. 229–237.

[21] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *CVPR*.

[22] Can Liu, Qiwei Zhong, Xiang Ao, Sun Li, Wangli Lin, Jinghua Feng, Qing He, and Jiayu Tang. 2020. Fraud Transactions Detection via Behavior Tree with Local Intention Calibration. In *KDD*. 3035–3043.

[23] Xu-Ying Liu, Jianxin Wu, and Zhi-Hua Zhou. 2009. Exploratory Undersampling for Class-Imbalance Learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 39, 2 (2009), 539–550.

[24] Yang Liu, Xiang Ao, Qiwei Zhong, Jinghua Feng, Jiayu Tang, and Qing He. 2020. Alike and Unlike: Resolving Class Imbalance Problem in Financial Credit Risk Assessment. In *CIKM*.

[25] Ziqi Liu, Chaochao Chen, Xinxing Yang, Jun Zhou, Xiaolong Li, and Le Song. 2018. Heterogeneous Graph Neural Networks for Malicious Account Detection. In *CIKM*.

[26] Zhiwei Liu, Yingtong Dou, Philip S Yu, Yutong Deng, and Hao Peng. 2020. Alleviating the Inconsistency Problem of Applying Graph Neural Network to Fraud Detection. In *SIGIR*.

[27] Amalia Luque, Alejandro Carrasco, Alejandro Martín, and Ana de las Heras. 2019. The impact of class imbalance in classification performance metrics based on the binary confusion matrix. *Pattern Recognition* 91 (2019), 216–231.

[28] Julian John McAuley and Jure Leskovec. 2013. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. In *WWW*.

[29] Jennifer Neville, Özgür Şimşek, David Jensen, John Komoroske, Kelly Palmer, and Henry Goldberg. 2005. Using Relational Knowledge Discovery to Prevent Securities Fraud. In *KDD*.

[30] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. PyTorch: An imperative style, high-performance deep learning library. In *NeurIPS*.

[31] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[32] Minlong Peng, Qi Zhang, Xiaoyu Xing, Tao Gui, Xuanjing Huang, Yu-Gang Jiang, Keyu Ding, and Zhigang Chen. 2019. Trainable undersampling for class-imbalance learning. In *AAAI*.

[33] Tahereh Pourhabibi, Kok-Leong Ong, Booi H Kam, and Yee Ling Boo. 2020. Fraud detection: A systematic literature review of graph-based anomaly detection approaches. *Decision Support Systems* (2020), 113303.

[34] Shebuti Rayana and Leman Akoglu. 2015. Collective Opinion Spam Detection: Bridging Review Networks and Metadata. In *KDD*.

[35] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. 2018. Learning to reweight examples for robust deep learning. In *ICML*.

[36] Min Shi, Yufei Yang, Xingquan Zhu, David Wilson, and Jianxun Liu. 2020. Multi-Class Imbalanced Graph Convolutional Network Learning. In *IJCAI*.

[37] Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. 2019. Meta-Weight-Net: Learning an Explicit Mapping For Sample Weighting. In *NeurIPS*. 1917–1928.

[38] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *ICLR*.

[39] Daixin Wang, Jianbin Lin, Peng Cui, Quanhui Jia, Zhen Wang, Yanming Fang, Quan Yu, Jun Zhou, Shuang Yang, and Yuan Qi. 2019. A Semi-supervised Graph Attentive Network for Financial Fraud Detection. In *ICDM*.

[40] Jianyu Wang, Rui Wen, Chunming Wu, Yu Huang, and Jian Xion. 2019. Fdgars: Fraudster detection via graph convolutional networks in online app review system. In *WWW*. 310–316.

[41] Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yu Gai, Tianjun Xiao, Tong He, George Karypis, Jinyang Li, and Zheng Zhang. 2019. Deep Graph Library: A Graph-Centric, Highly-Performant Package for Graph Neural Networks. *arXiv preprint arXiv:1909.01315* (2019).

[42] Wentao Wang, Suhang Wang, Wenqi Fan, Zitao Liu, and Jiliang Tang. 2020. Global-and-Local Aware Data Generation for the Class Imbalance Problem. In *SDM*.

[43] Xiao Wang, Yuanfu Lu, Chuan Shi, Ruijia Wang, Peng Cui, and Shuai Mou. 2020. Dynamic Heterogeneous Information Network Embedding with Meta-path based Proximity. *IEEE Transactions on Knowledge and Data Engineering* (2020).

[44] Xiao Wang, Meiqi Zhu, Deyu Bo, Peng Cui, Chuan Shi, and Jian Pei. 2020. AM-GCN: Adaptive Multi-channel Graph Convolutional Networks. In *KDD*. 1243–1253.

[45] X. Yin, Xiang Yu, Kihyuk Sohn, Xiaoming Liu, and M. Chandraker. 2019. Feature Transfer Learning for Face Recognition With Under-Represented Data. In *CVPR*.

[46] Chong You, Chi Li, Daniel Robinson, and René Vidal. 2018. A Scalable Exemplar-Based Subspace Clustering Algorithm for Class-Imbalanced Data. In *ECCV*.

[47] Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. 2020. Graphsaint: Graph sampling based inductive learning method. In *ICLR*.

[48] Tong Zhao, Chuchen Deng, Kaifeng Yu, Tianwen Jiang, Daheng Wang, and Meng Jiang. 2020. Error-Bounded Graph Anomaly Loss for GNNs. In *CIKM*. 1873–1882.

[49] Qiwei Zhong, Yang Liu, Xiang Ao, Binbin Hu, Jinghua Feng, Jiayu Tang, and Qing He. 2020. Financial Defaulter Detection on Online Credit Payment via Multi-view Attributed Heterogeneous Information Network. In *WWW*.